# How to Draw Commands? An Elicitation Study for Sketching on Spreadsheets


Marc Hesenius
marc.hesenius@uni-due.de
University of Duisburg-Essen
Essen, Germany

Mak Krvavac
mak.krvavac@stud.uni-due.de
University of Duisburg-Essen
Essen, Germany

Valbjörn Jón Valbjörnsson
valbjorn@hi.is
University of Iceland
Reykjavík, Iceland

Theresia Mita Erika
tme1@hi.is
University of Iceland
Reykjavík, Iceland

Matthias Book
book@hi.is
University of Iceland
Reykjavík, Iceland

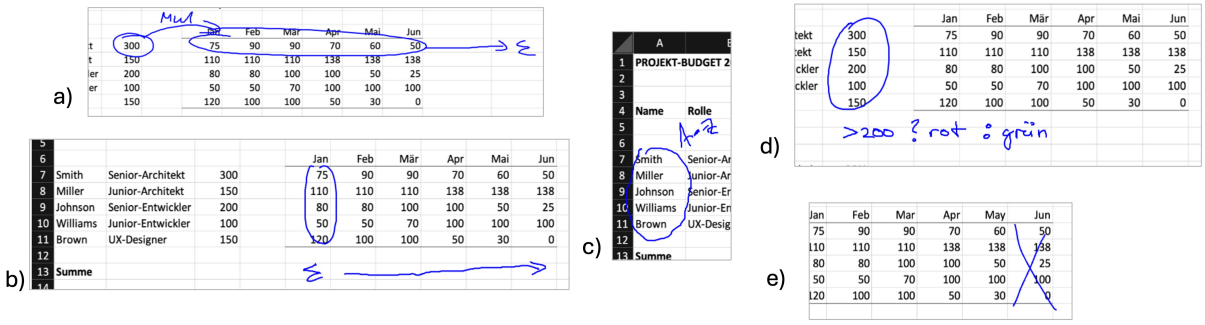




**Figure 1: Proposals for sketching commands on spreadsheets: a) Multiplying a cell with the sum of other cells; b) Calculating multiple sums; c) Sorting a column alphabetically; d) Applying conditional formatting to cells; and e) Deleting a table column**


## Abstract

Sketching is one of the oldest techniques humans use to express themselves. We sketch to visualize concepts, externalize memory, and communicate ideas. However, we barely use sketching to interact with computers. Given how naturally sketching comes to humans, we believe untapped potential exists in being able to simply draw commands onto a user interface. In this paper, we present results of an elicitation study about expressing common operations in spreadsheets through sketching. Spreadsheets are an interesting class of applications because they are widely used, support complex data and operations, and are available on touch-enabled devices. Our results show that despite considerable variation in syntactic details, participants gravitate towards recurring patterns (e.g., enclosures and arrows, examples and cross-references, and temporal sequences of strokes). The sketch patterns we identified can be a first step towards developing interpreters of sketched commands, and thus enable new means of interacting with spreadsheets and other applications.




*CHI '25, Yokohama, Japan*



ACM ISBN 979-8-4007-1394-1/25/04
https://doi.org/10.1145/3706598.3715269



## CCS Concepts

• **Human-centered computing** → **Interaction techniques**; **Touch screens**; **Empirical studies in HCI**; • **Software and its engineering** → *Programming by example.*

## Keywords

Spreadsheets, Sketching, Interaction Techniques, Pens, Touchscreens

**ACM Reference Format:**
Marc Hesenius, Mak Krvavac, Valbjörn Jón Valbjörnsson, Theresia Mita Erika, and Matthias Book. 2025. How to Draw Commands? An Elicitation Study for Sketching on Spreadsheets. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 31 pages. https://doi.org/10.1145/3706598.3715269


## 1 Introduction

Sketching – the creation of informal, ad-hoc drawings – comes naturally to humans and is one of the oldest communication techniques we employ, second only to speech in order to express ourselves from a young age [9]. We sketch to organize and externalize our thoughts [42], and to communicate concepts and ideas to our peers [6], whether in the arts, sciences, or in engineering disciplines. Sketching also is an essential part of human-computer interaction (HCI): Most design teams (about 97% [24]) sketch in the early phases of development projects, and dedicated methods exist for collaborative informal sketching (e.g., [14, 16, 32, 49, 52]).

Digital pens or styluses have been an accessory to mobile devices since the release of the first personal digital assistant (PDA) about

30 years ago, but have rarely been used as more than a precise pointing device, an emulated paintbrush, or a tool for handwriting input. Given how naturally sketching comes to humans when expressing plans and concepts, it is surprising that it is far less common to express commands through sketching than through keyboard shortcuts, mouse clicks, or touch gestures. Although the technical means for sketching on interactive surfaces exist, only few approaches leverage such input to express commands (see Section 2).

We therefore hypothesize there is untapped potential in enabling users to interact with applications by drawing directly on their graphical user interface (GUI) with digital ink, in similar ways as copy editors annotate printed manuscripts in order to express desired changes.

### 1.1 Sketching on Spreadsheets

We focus on spreadsheets as a particularly interesting case for sketching: They are commonly used by many people [33], can be quite complex in terms of structure and possible operations [5], lend themselves to direct manipulation [2], and are available on touch-enabled devices that can be operated with a digital pen.

However, spreadsheets often display a lot of data and thus require a lot of screen real estate that, on mobile devices, is severely encroached upon whenever users need to use the on-screen keyboard. We argue that simply sketching commands on the GUI would yield a better user experience, because it allows users to interact more naturally and immediately with their data than through complex menus and the disruptive on-screen keyboard.

As a first step towards enabling interaction with spreadsheets through sketching in ways that go beyond mere handwriting recognition and visual annotation, we conducted an elicitation study in which we asked participants to express a variety of common spreadsheet operations by sketching on top of an application's GUI (see Figure 1 for some examples). The insights from our study can serve as a starting point for understanding how users conceptualize and visually express complex operations, what variety and commonality exists in those sketch expressions, and how they can be recognized and interpreted by the user interface (UI) logic of sketch-enabled spreadsheets, and potentially other classes of software applications.

### 1.2 Goals and Contributions

Our study was guided by the following research questions (RQ):

- How would users sketch common spreadsheet operations? (RQ1) Specifically, how would they:
  - edit data in particular cells? (RQ1.1)
  - manipulate the spreadsheet's structure? (RQ1.2)
  - format data in particular cells? (RQ1.3)
  - visualize data in the spreadsheet? (RQ1.4)
  - perform arithmetic operations? (RQ1.5)
- What larger patterns and strategies do users employ in sketch expressions that may be relevant for their automated interpretation? (RQ2)

As detailed in Section 3, we presented 36 participants with 20 referents, asked them to express the respective operations through sketching, categorized their proposals for the calculation of agreement rates, and identified emerging patterns. Our results (Section 4) show that despite considerable syntactical variation in the visual expression of commands, many participants tended to gravitate to conceptually similar patterns of sketch expressions. As we discuss in Section 5, our participants' proposals provided insights into larger patterns such as

- how participants used different visual expressions for selecting and connecting elements of their sketch and/or elements of the application's GUI;
- how participants expressed repetition of commands and continuations of series in spreadsheets;
- how participants sketched examples to specify desired outcomes of commands;
- how participants leveraged drawing styles, table structures, and the temporal sequence of sketch elements to imply information that was not sketched explicitly;
- when participants fell back to preferring GUI interactions over sketching commands.

Our contribution to the field of HCI in general and the field of interacting with interactive surfaces in particular is thus threefold:

- an analysis of agreement across the sketched expression of 20 common spreadsheet commands by 36 study participants;
- specific patterns for the sketched expression of five fundamental classes of spreadsheet commands, and general sketching strategies derived from them; and
- insights into requirements for the sketch recognition and command interpretation components of future applications.

## 2 Related Work

Our study builds upon four aspects: Using sketching as an interaction modality (Section 2.1), expressing commands through sketches (Section 2.2), interacting with spreadsheets on mobile devices (Section 2.3), and conducting elicitation studies (Section 2.4).

### 2.1 Sketching as an Interaction Modality

Sketching can be understood as an interaction on a wide range of timescales, where the interval between creation and interpretation of the sketch influences which aspect of the sketch carries most information.

Some paper sketches may not be revisited, revised or responded to until *days, weeks or months* after they have been drawn – either by the author themselves, or by recipients unknown to the author. Dix and Gongora [8, p. 1] describe sketching as an artifact in a form of externalization, the "active shaping of the world as an intellectual resource", where sketches, alongside models and prototypes, represent an abstraction of a product and serve as a form of communication with clients or colleagues. They emphasize how externalizations, and thus sketches, serve as a means of interaction with the world (e.g., through an architectural model), with others (e.g., through a diagram), and even with oneself (e.g., by taking notes). In HCI design, for example, sketches are at the very core of moving from an idea to a particular solution [3, 11], with dedicated courses and tutorials existing on how to apply sketching when developing a suitable user experience (e.g., Marquardt [20] at CHI'17). For such long-term consumption by potentially unknown audiences, the sketch needs to be refined to the level of an almost self-explanatory *artifact*, or be embedded in rich persistent context.

In collaborative work on a whiteboard, sketches become part of a context-laden conversation, being perceived and evolved on a timescale of *seconds or minutes*. Such a conversational sketch might contain minimal explicit information, but rely heavily on the ephemeral *context* established between the interlocutors who are using the sketch to support and focus their conversation [6, 19].

When drawing with a digital stylus in a software tool (such as the early InkKit [7] and its many successors), sketching becomes an almost *real-time* activity: Beyond reacting immediately to some strokes or shapes (whether by just displaying digital ink, or by, e.g., highlighting underlying content to indicate the extent of a selection), the application might wait for the user to complete a more complex shape or sketch in order to interpret and respond to it as a whole. In such real-time interaction, the *process* of sketching (i.e., the particular spatial arrangement and temporal sequence of strokes in relation to some substrate) may carry just as much information as its final outcome, the sketch.

Touch gestures [55] are at the extreme end of this spectrum, as they typically do not involve the creation of a visual artifact at all, but transport all their information in their performance. However, compared to a fingertip, a stylus producing digital ink enables a much higher precision when relating shapes to each other and the underlying content, allowing the user to create more detailed, complex, and substrate-related expressions of intended commands through sketching than through gesturing [41].

## 2.2 Sketches as Command Expressions

Sketches have been proposed as a means to convey user intentions in a variety of domains and forms. In DataInk [51], sketching is closely integrated into an iterative design process that interleaves illustration, layout and data binding to define interactive data visualizations. Similarly, Doodle2App [21] lets users sketch GUI widgets in order to generate working GUI prototypes, and PSDoodle [22] allows users to sketch a GUI in order to search for similar interface designs. More recently, Lee et al. [17] have compared sketch-based and text-based prompts for generative artificial intelligence models, and describe how both influence the design process of real-world objects. Such uses of sketching to *parameterize* operations are convenient when the parameters are difficult or tedious to express by other means, e.g., verbally.

Other approaches use sketches to *trigger* operations. Taking inspiration from the traditional process of writing and revising musical notation on paper, Musink [39] uses an Anoto pen to bridge the gap between paper and software. The system enables users to define their own handwritten annotations of musical notation on paper, and associate those gestures with commands to be executed by a music composition tool. This approach is notable both for employing user-defined sketches to trigger commands, and its integration of physical and digital media.

KnottyGestures [40] establishes a similar connection between paper-based content and digital operations on that content. Using a Livescribe pen, users can add small circles as interactive decorations to their pen strokes on paper. These "knots" serve as custom, in-situ UI elements for simple operations such as controlling the audio playback or interacting with written content (such as performing mathematical operations on tabular data).

In the domain of data analysis, SketchSliders [38] enables users to interact with data visualizations by sketching GUI elements that they would like to use to analyze data on the fly. By creating their own "data controllers", users gain freedom to tailor the interaction with the data to their needs at runtime.

In a similar vein, ActiveInk [29] allows users to examine data interactively, enabling them to sketch on data visualizations in order to perform actions like isolating (cutting) parts of data for further processing. It thus shows how sketching can be used to externalize thoughts during data analysis. However, the externalization behavior of users depends on what analysis stage they are in, and thus requires consideration during application design [15].

CodePad [25] allows software developers to interact with source code in an integrated development environment (IDE) through sketches drawn on a peripheral screen. Similarly, Raab [28] and Samuelsson and Book [30] explore how sketching can be used to interact with an IDE by drawing on top of source code.

Transferring these approaches to spreadsheets, we pursue the same goal of using sketching as a command-expression modality: We hypothesize that users might find it intuitive to sketch on a digital spreadsheet like they would on a printed table, e.g., using lines, arrows, and enclosures, in order to express desired modifications, calculations, and other operations – especially in the absence of traditional input modalities such as a mouse and keyboard.

## 2.3 Interaction with Spreadsheets

Spreadsheets have been reported to be hard to comprehend [36], which, among other things, is due to their visual expansiveness, the amount of data contained in them, and the conceptual and formulaic relationships between their cells. Spreadsheets have therefore received plenty of attention from the HCI community, e.g., to simplify formula programming with natural language [35]. For the sake of brevity, we will focus our review on gesture-based interaction on mobile devices and the use of pens for sketching.

Perelman et al. [26, 27] propose to use a smartphone as an extension for a tablet to decouple the command layer from the visualization layer. Others have introduced gestures into spreadsheets, which we consider closely related to sketches as they build upon a similar kind of spatio-temporal input. Burnett and Gottfried [2], for example, introduce self-defined gestures into spreadsheets to create and manipulate graphical objects, and Takayama et al. [37] elicited mid-air gestures for operations such as inserting, deleting and sorting cells.

Other works have introduced basic sketching to simplify spreadsheet creation. With Tableur [54], users can quickly sketch outlines of spreadsheets and use a gesture to *solidify* the sketch into a spreadsheet. The tool also allows users to write simple formulae that are converted with the help of handwriting and math recognition into executable parts of the spreadsheet. Gesslein et al. [10] move the interaction into virtual reality, motivated by the availability of unlimited screen real estate, and propose to use a (real) pen that is mapped into the virtual reality to interact with spreadsheets.

However, the focus of these approaches remains on using the pen as an alternative for touch input, handwriting and menu interaction, not for expressing commands in immediate relation to the spreadsheet contents.

Cavez et al. [4] argue that spreadsheets present a unique challenge for interaction design as they consist of three layers – grid, value, and navigation – that interfere with each other when it comes to user input. As a consequence, even simple tasks such as selecting elements can become frustrating. Based on an elicitation study, they introduce dedicated interaction modalities for different aspects: Using the pen to select, single-touch to manipulate, and multi-touch to navigate. Most notably, they use pen input for fine-grained selection not only on the grid level (i.e., single cells), but also on the value level (i.e., parts of a value). Their study is focused on different direct manipulation tasks, emphasizing what users could be capable of if application providers would implement all possibilities. Some of their findings are also reflected in our data, e.g., that users commonly employ encircling and underlining for selection, but our study extends the use of pens beyond selection and introduces sketching as an alternative interaction modality enabling users to express a broad variety of complex commands.

## 2.4 Elicitation Studies and Agreement Measures

Elicitation studies – studies where participants are presented with a system's action and have to propose a suitable interaction – have become exceptionally popular in recent years, especially in the area of gesture-based applications. Since their introduction in 2005 as a *guessability study* [47] and the first incarnation in the area of gesture-based interaction by Wobbrock et al. [48], a plethora of elicitation studies have been conducted in areas as diverse as eliciting gestures for police officers interacting with drones [13], analyzing differences in gestures performed by different age groups [12], identifying mid-air gestures for interaction with spreadsheets [37], and interacting with source code through sketching [30, 31]. Villarreal-Narvaez et al. [45] reviewed 216 gesture elicitation studies, emphasizing their popularity, but also highlighting several concerns, especially regarding participant selection.

Vatavu and Wobbrock [44] describe in general how elicitation studies should be evaluated with regard to calculating agreement measures under different circumstances, and how to perform subsequent statistical evaluations. Our approach to analyze the gathered data follows their recommendations (see Section 3.5).

# 3 Method

To answer our research questions, we conducted an *elicitation study*, inspired by approaches for identifying suitable gestures with users. While alternatives, e.g., an exploratory study, would also have been suitable, we opted for an elicitation study to benefit from the established analytical framework and the similarities of gesture- and sketch-based applications. Furthermore, we were curious about similarities in responses, whose analysis is at the core of elicitation studies. In the following subsections, we describe our participant cohort (Section 3.1), the referents we prompted participants to propose sketches for (Section 3.2), the application used for data collection (Section 3.3), the setup and process of the elicitation sessions (Section 3.4), and our approach to coding and categorizing the participants' proposals (Section 3.5).

## 3.1 Participants

We recruited 36 volunteers; 18 from Germany and 18 from Iceland. Four additional participants (two from either country) piloted the study to test and refine the methodology and phrasing of tasks, but their responses were removed from the result set.

A key criterion in selecting the participants was their prior experience with spreadsheets. All participants confirmed having at least basic knowledge and some practical experience with using spreadsheets. While some participants had extensive and long-term experience, others' spreadsheet experience was limited to basic functions.

One participant indicated having experience with using spreadsheets on a tablet, and three participants indicated having used spreadsheets on other touch-enabled devices such as smartphones. Twenty-four participants reported owning a private tablet, 11 of which mainly use their tablet for surfing the Internet. Seven indicated that they predominantly use their tablet for work, while others use it for gaming, reading books, and consuming other media (two each). Fifteen participants indicated they had experience with using a pen or stylus on a tablet.

Seven participants were 18–24 years old, 18 were 25–30 years old, and 11 participants were 30–40 years old. Twelve participants were female, 23 were male, and one identified as non-binary. Three participants were left-handed, and 33 were right-handed.

The study was approved by the ethics committee of the Faculty of Business Administration and Economics at the University of Duisburg-Essen in September 2023. All participants consented to the anonymous collection of ink strokes, survey responses, and screen recordings. Participants did not receive any compensation and could withdraw from the study at any time.

## 3.2 Tasks

We asked each participant to solve a set of 20 tasks involving common spreadsheet operations (see Table 1). We designed the tasks to cover five different types of operations:

- *Data Edit (RQ1.1)*: Operations to change cell contents, e.g., entering or deleting values (Tasks 1, 2, 3, and 14).
- *Structure Change (RQ1.2)*: Operations to modify the structure of the spreadsheet, e.g., inserting, transposing, and sorting cells (Tasks 5, 6, 7, 15, and 16).[1]
- *Formatting (RQ1.3)*: Operations to modify the visual presentation of cell content, e.g., framing or coloring cells (Tasks 9, 10, 11, and 19).
- *Visualization (RQ1.4)*: Operations to govern graphical representations, e.g., creating charts (Tasks 12 and 13).[2]
- *Calculation (RQ1.5)*: Arithmetic operations that involve the contents of several cells, such as adding or multiplying values (Tasks 4, 8, 17, 18, and 20).

[1] One might argue that the distinction between modifying cell contents and altering spreadsheet structure is fluid. For example, sorting and transposing cells may be seen as either localized value changes or broader structural modifications. We classified operations as "data edits" when they would change the data, while operations that merely changed the data's organization but not their values as "structural changes".
[2] Formatting and visualization are related but distinct: formatting concerns the typographic presentation of raw values, while visualization represents values in non-alphanumeric forms like charts.

**Table 1: Elicitation study tasks (annotations in [ ] indicating action and parameter components were not shown to participants)**

| Task | Title | Prompt | Task Type / RQ | Group |
|---|---|---|---|---|
| 1 | Change value | Enter the value [A] 100 [$P_1$] into cell F9 [$P_2$]. | Data Edit (RQ1.1) | W |
| 2 | Delete value | Delete the contents [A] of cell E8 [$P_1$]. | Data Edit (RQ1.1) | W |
| 3 | Delete values | Delete the contents [A] of cells J7–J11 [$P_1$]. | Data Edit (RQ1.1) | A |
| 4 | Add two values | Calculate the sum [A] of cells I8 [$P_1$] and J8 [$P_2$] in cell K8 [$P_3$]. | Calculation (RQ1.5) | A |
| 5 | Insert column | Insert a new column [A] between columns B and C [$P_1$]. | Structure Change (RQ1.2) | A |
| 6 | Remove column | Remove the whole column [A] B [$P_1$] from the table. | Structure Change (RQ1.2) | A |
| 7 | Move cells | Move [A] cells A15–C15 [$P_1$] to H3 [$P_2$]. | Structure Change (RQ1.2) | A |
| 8 | Add multiple values | Calculate the sum [A] of cells E7–E11 [$P_1$] in cell E13 [$P_2$]. | Calculation (RQ1.5) | B |
| 9 | Format values | Format the values [A] in cells C7–C11 [$P_1$] as Dollar amounts with two decimal places [$P_2$]. | Formatting (RQ1.3) | B |
| 10 | Frame cell | Add a single frame [A] around cells A15–C15 [$P_1$]. | Formatting (RQ1.3) | B |
| 11 | Transfer formatting | Transfer the formatting [A] of cell A13 [$P_1$] to cells E13–J13 [$P_2$]. | Formatting (RQ1.3) | B |
| 12 | Create pie chart | Create a pie chart [A] from the values in cells E8–J8 [$P_1$]. | Visualization (RQ1.4) | B |
| 13 | Create bar chart | Create a bar chart [A] from the values in cells C7–C11 [$P_1$]. | Visualization (RQ1.4) | B |
| 14 | Continue series | Continue the series of values [A] in row 6 (E6–J6) [$P_1$] up to cell M6 [$P_2$]. | Data Edit (RQ1.1) | B |
| 15 | Transpose values | Transpose [A] the names in cells A7–A11 [$P_1$] to row 17 [$P_2$]. | Structure Change (RQ1.2) | B |
| 16 | Sort data | Sort the data [A] in rows 7–11 [$P_1$] alphabetically [$P_2$] by the employee names in column A [$P_3$]. | Structure Change (RQ1.2) | C |
| 17 | Calculate multiple sums | For each month, calculate the sum [A] of all employees' hours [$P_1$] in cells E13–J13 [$P_2$]. | Calculation (RQ1.5) | C |
| 18 | Construct formula | For each employee, calculate [A] the gross hourly rate in column D [$P_3$] by factoring the overhead cost percentage from cell C15 [$P_2$] into the net hourly rate in column C [$P_1$]. | Calculation (RQ1.5) | C |
| 19 | Conditional formatting | Define a conditional formatting rule [A] for cells C7–C11 [$P_1$] so values greater than 200 [$P_2$] are displayed in red but others in green [$P_3$]. | Formatting (RQ1.3) | C |
| 20 | Multiply values | Calculate [A] the total salary of employee Smith *[by multiplication of cell C7 [$P_1$] with the sum of cells E7–J7 [$P_2$]]* in cell L7 [$P_3$]. | Calculation (RQ1.5) | C |

These tasks were chosen to represent a broad variety of operations, including commands that are:

- common in many applications (e.g., Task 1), or spreadsheet-specific (e.g., Task 15)
- variations of the same operation (e.g., Tasks 2, 3, and 6)
- referring to different aspects of a spreadsheet – its contents (e.g., Task 4), its grid (e.g., Task 5), its formatting (e.g., Task 10), or its semantics (e.g., Task 14)
- tied to concrete visual anchors (e.g., Task 7), or rather abstract (e.g., Task 19)
- traditionally GUI-centric (e.g., Task 9), keyboard-centric (e.g., Task 20), or mouse-centric (e.g., Task 14)
- varying in complexity (from Task 1 to e.g., Task 18)

Despite this variety, we strove to include only tasks that we assumed most participants would understand without further explanation, and refrained from including more tasks in order to avoid participant fatigue. We consider the resulting task set as a starting point that can help us identify which types of commands would merit deeper examination – for example, to explore how participants would express different types of formulae, or how they would deal with challenges such as references to off-screen cells, highly parameterizable operations etc.

To familiarize participants gently with the idea of expressing commands through sketches, we grouped the tasks according to their difficulty. Two almost-trivial tasks (Group W) were used to help each participant "warm up" for their session. The remaining tasks were assigned to three groups (A, B, C) based on our subjective impression of their difficulty. We considered tasks more difficult if they involved abstract concepts (e.g., Task 11), uncommon operations (e.g., Task 15), or many parameters (e.g., Task 20). These difficulty groups were used to roughly align the progression of tasks that participants saw to their expected learning curve and increasing familiarity with sketching, as detailed in Section 3.4.

All tasks referred to the same spreadsheet (Figure 2) showing a simple budget calculation for a fictional IT project. We used the same spreadsheet with a common scenario for all tasks in order to minimize participants' mental effort in orienting themselves in the spreadsheet, and to help them focus on conceiving suitable sketches for the shown referents. Participants from Iceland worked with an English version of the spreadsheet and tasks while participants from Germany used a translated version.

## 3.3 Application

Rather than letting participants sketch on a familiar, working spreadsheet application, whose complex GUI and live feedback would have been distracting, we presented our participants with a static image of a spreadsheet, and asked them to show us how they would sketch on this spreadsheet if it was a live application. We instructed participants to assume that all their sketches could be automatically interpreted correctly by the hypothetical spreadsheet application, in order to discourage them from censoring their ideas to accommodate any imagined technical properties or limitations of the sketch interpretation component.

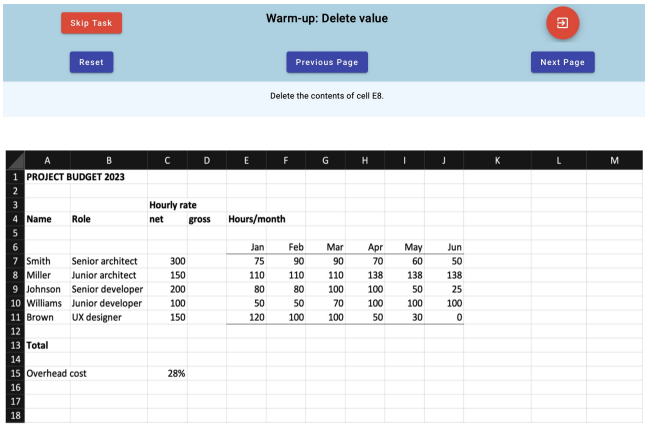



**Figure 2: Screenshot of the English survey app with navigation bar, task prompt, and sketch area with static spreadsheet**

To collect all relevant data, we developed a survey app[3] specifically designed to conduct elicitation studies. For each task, the app displays a referent above the static spreadsheet image (as shown in Figure 2) and records the participant's sketch proposal in the form of digital ink strokes, as well as (given consent) their audio as they think aloud about their sketch. In addition, the app collects demographic information and feedback on each sketch.

Participants were free to navigate the app without help from the experimenter. Using the buttons above the spreadsheet, they could move between tasks, discard their current sketch via the Reset button to start over with a task, skip a task, or exit the study entirely. Since our app did not feature an eraser tool, participants typically used the Reset button to start over when they had misdrawn something. No participants used the opportunity to skip a question or exit the study mid-way.

The survey app was implemented as a web application using the Angular framework, allowing us to run the survey on a variety of devices such as tablets, touch-enabled laptops, or wall-mounted interactive displays. While we may use this cross-platform compatibility for future studies, all participants in this study used a tablet with a digital pen.

## 3.4 Sessions

We met with each participant in an individual session. Except for the experimenter and the participant, no further observers were present during each session. All sessions lasted approximately 25 to 35 minutes and were led by experimenters M.K. or T.M.E. in German or English language, depending on the participant's choice. Each session was divided into three phases:

(1) *Familiarization.* We provided participants with a tablet with the survey app and a digital pen, and verbally introduced them to the research objective and the idea of sketching as an interaction modality in general. We also explained the study process and the collected data. We furthermore encouraged participants to articulate their thoughts at any time (*think-aloud*), particularly regarding the rationale for their sketch solutions, and any issues they encountered. Participants were at this point asked whether they consented to participate in the study, and could only proceed after acknowledging the consent form and information sheet displayed in the app. Subsequently, we explained the UI of the survey app, and participants were given the opportunity to familiarize themselves with the UI by toying around with the pen under the experimenter's guidance. Once they were ready, participants could start the study themselves and navigate it independently.

(2) *Sketching tasks.* In the study's main part, the survey app showed participants a sequence of 20 tasks that they could navigate freely. To accommodate participants' learning curve and increasing familiarity with sketching as the study progressed, while at the same time avoiding biases potentially introduced by the order of tasks, the survey app showed the task groups in order of increasing difficulty (W, A, B, C), but randomized the order of tasks within each group. The experimenter would assist the participant in case they struggled to understand a referent, but did not provide suggestions on how to respond to it. No time constraints were imposed, allowing participants to control the study's pace themselves. Once a participant completed a task and proceeded to the next one, an intermediate questionnaire asked them to rate the perceived difficulty of the task and their satisfaction with their sketch.

(3) *Demographic survey.* After the final task, the survey app prompted the participant to provide demographic information regarding their age and familiarity with tablets and spreadsheets. Recorded data was stored in pseudonymized form.

A common problem in elicitation studies is *legacy bias*, i.e., the influence of experience with existing applications and established HCI techniques. Morris et al. [23] propose three techniques to counter legacy bias: priming (focusing on the possibilities of the interaction technique), production (creating multiple solutions), and partners (teaming up with a partner). We focused on *priming* during the sessions to stimulate the participants' creative ideas and specifically instructed them to distance themselves from traditional mouse and keyboard interaction. We also deliberately conducted the study on a tablet rather than a pen-enabled laptop, so participants would not expect to have a mouse and keyboard available, and we used a screenshot of a spreadsheet without GUI widgets, so participants would not be drawn to familiar buttons and menus. We also discussed sketching as a means of human interaction, aiming to give participants a clearer idea of what sketching entails. Furthermore, we allowed participants to generate multiple proposals and reset their current sketch at any time, supporting *production* as a strategy to mitigate legacy bias.

## 3.5 Analysis

To identify patterns in our participants' sketched proposals, and to analyze if certain patterns were favored, we calculated agreement rates for each task's proposals, as described by Vatavu and Wobbrock [44]. To define what constitutes agreement, we expanded

[3]Publicly available at https://github.com/mhesenius/elicitation-ui-web

upon an approach inspired by Shneiderman's Object-Action principle [34]: Like any interaction modality, sketching is used to express *commands*, i.e., directives to perform specific operations. We conceptually break commands down into *actions* and *parameters*, and then categorize participants' proposals according to how they visually expressed those command aspects. For example, the command "Move cells A15–C15 to H3" (Task 7) comprises the action "move" with the range A15–C15 as parameter $P_1$ ("what?") and the cell H3 as parameter $P_2$ ("where?"). Samuelsson and Book [31] suggested a taxonomy of elements that can be used to encode how participants express actions and parameters in a sketched command. We adapt and extend their taxonomy as follows to encode the proposals that our participants produced:

- **TXT:** Handwritten alphanumeric characters including punctuation and intermittent symbols such as arithmetic operators, e.g., "Σ(E7,E11)".
- **TXD:** Decorated TXT, i.e., handwriting with additional markings or embellishments such as underlining or enclosures, e.g., "SORT" or "®".
- **SYM:** Freestanding symbols to which participants ascribe meaning, e.g., "?" or "▹". Arithmetic operators such as "+" were coded as SYM when standing on their own, but as TXT when part of a formula (e.g., "I8+J8").
- **CON:** Connections such as lines or arrows between cells, GUI widgets, and/or sketched shapes. Arrows were only coded as CON when their start- and endpoints carried meaning, but as SYM when the locations of their start- and endpoints were irrelevant. We designate the start and end of a connection as CON.START and CON.END.
- **SEL:** Selections of one or more cells, e.g., by encircling or underlining. Such shapes were coded as selections (SEL) when applied to spreadsheet content, but as decorations (TXD) when applied to handwriting.
- **EXA:** Sketched examples of a desired command outcome or parameter, such as "$123,00" to describe the currency format in Task 9, or a pie chart to describe the visualization in Task 12. While one might argue that such sketches could also be coded as TXT or SYM, we considered the drawing of examples to be such a sketch-specific technique that cannot be as easily integrated in other input modalities that we felt it warranted its own code, expanding the taxonomy of Samuelsson and Book [31].
- **LOC:** Locations of sketch elements whose placement carries meaning. For example, if a participant wrote "100" into cell F9 in response to Task 1, the "what" parameter of the "entry" action is expressed by the handwritten digits (coded as "$P_1$:TXT"), while the "where" parameter is expressed by the location of those digits (i.e., $P_2$:LOC.$P_1$).
- **GUI:** Non-sketch interactions with GUI elements, such as taps on buttons or menus. While some participants sketched GUI widgets in response to tasks, the audio recordings and researchers' notes made clear that they were just illustrating what they expected to see, but did not consider those sketches as command input. This code corresponds to the NON code used by Samuelsson and Book [31], which we renamed to GUI because all non-sketch interactions we observed involved GUI elements.
- **IMP:** Implied information, i.e., any action or parameter that was not explicitly sketched but could be inferred from other sketch elements or context. For example, if a participant crossed out the contents of cell E8 in response to Task 2, we considered the "delete" action to be implied by the style of the selection used to express the "where" parameter $P_1$.
- **MIS:** Missing information, i.e., any action or parameter that was not explicitly sketched and could not be inferred from other sketch elements or context. Since we instructed participants to assume the hypothetical spreadsheet application could easily interpret any of their sketches, we were quite generous in considering not explicitly sketched command aspects as IMP, unless there was no way of plausibly deriving them from the existing sketch.
- **SKP:** Skipped tasks that a participant declined to provide a proposal for. (This did not occur in our study.)

After completing the sessions with all 36 participants, we reviewed all 720 responses and coded them according to our taxonomy. Where participants had reset a task and started over with their sketch, we disregarded obviously incomplete sketches and only evaluated their final, complete sketches.[4] The coding was based on the screenshot of each proposal, if its interpretation was obvious and in line with the task. In other cases (e.g., unclear meaning, command aspects not expressed explicitly, or any other uncertainties), we reviewed the screen and audio recordings for clarification.

The collected proposals were initially roughly clustered into visually similar groups by one author (M.K.), and then coded according to the aforementioned taxonomy by two other authors independently (M.H. and M.B.). Their codings were subsequently compared and discussed in detail. While mostly consistent, several discrepancies had to be resolved and aligned across the whole set of responses. Another author (V.J.V.) finally reviewed the aligned codings and the resulting categories for consistency.

To define what constitutes "agreement" for the purpose of our analysis, we generally assigned proposals to the same category if they received the same coding across all command aspects (i.e., if the actions and all parameters were expressed in the same way). However, in cases where the process for interpreting the sketch would be similar despite a participant using a slightly different visual syntax, we combined several codes into one category.

In the interest of reflecting only those differences between participants' proposals in the agreement rates that are actually relevant to sketching, we took the following additional steps in assigning the proposals for each referent to categories:

- All proposals involving a GUI code were assigned to a single category, regardless of differences in their non-GUI command aspects. This was done so that high levels of disagreement between proposals that did not use sketching (i.e., did not follow our instruction to express the command through sketching) would not obscure the level of agreement between sketched proposals. However, we did not exclude the

[4]Participants produced only one valid proposal for almost all referents. Only in rare cases did a participant produce two distinct complete sketches for a task. If the audio recording confirmed that the participant considered both sketches as viable alternatives, we included both of them in our analysis.

GUI-reliant proposals from the agreement rate calculation entirely, but included them as a single category, so those participants' choice *not* to pursue a sketch-based solution would remain represented as one variant in the agreement rate.
- Proposals involving missing command aspects (i.e., a MIS code) were removed from the agreement rate calculation, as they did not constitute a valid command. This also applied to the rare cases where participants had obviously misunderstood a referent.
- In the few cases where one participant offered two valid proposals for a referent, we included both of them in their respective categories.

Given these simplifications, we are confident that the agreement rates we report below reflect the diversity in how our participants actually sketched valid commands, without being biased by the diversity in invalid or non-sketch-based proposals.

## 4 Results

We used the categories that we obtained through the classification process described in the previous section to calculate *agreement rates (AR)* according to the method of Vatavu and Wobbrock [43]. Following the guidelines of Vatavu and Wobbrock [44], we chose AR over other agreement measures for the following reasons:

- While the agreement rates of most tasks were calculated based on a total of 36 proposals per referent, some were based on a slightly lower (if invalid proposals were excluded) or slightly higher total (if participants offered multiple valid proposals), as explained in the previous section. We chose AR over agreement scores (A) to ensure that our measure of agreement was comparable between referents despite this variation in the number of proposals.
- Our taxonomy of sketch expressions for command aspects provides a straightforward way of coding proposals based on clearly distinguishable visual cues. We therefore chose AR over consensus (C) as a measure of agreement.

For each task $t$, we calculated the proposals' agreement rate $a_t$ as denoted in Equation 1, where $S_t$ is the set of proposals for task $t$, and $S_{t,c}$ is a subset of proposals within $S_t$ that were assigned to the same category $c$ through our coding process. The AR can range from 0 (total disagreement) to 1 (total agreement). Figure 3 shows the resulting AR of all tasks.

$$a_t = \frac{\sum_{S_{t,c} \subseteq S_t} \frac{1}{2}|S_{t,c}|(|S_{t,c}| - 1)}{\frac{1}{2}|S_t|(|S_t| - 1)} \quad (1)$$

Vatavu and Wobbrock [44] aimed to determine the most precise test across multiple statistical tests to validate agreement scores. In their results, the percentile bootstrap [46] had the lowest error rate, and as result, they recommend it for use in elicitation studies. Vatavu and Wobbrock [43] also distinguish between low (< 0.1), medium (0.1–0.3), high (0.3–0.5), and very high (> 0.5) agreement. Although such classifications are valuable for interpreting elicitation results and provide a tangible metric, they should not be used to draw definitive conclusions or compare to other studies using ARs [44]. Several of our ARs are classified as "high". In accordance with the recommendations of Vatavu and Wobbrock [44], we used the bootstrap percentile test to calculate the mean AR and confidence intervals of our results with 95% confidence level. The test resulted in a mean AR of 0.2888, with lower and higher bounds of 0.245 and 0.338, respectively (Figure 4). This suggests that our mean AR is classified as "medium", although quite close to high agreement.

For each task, Table 2 shows its AR as well as the breakdown of the proposals into categories, their codes and sizes (i.e., the number of agreeing proposals in each category). For the sake of brevity, this table lists only categories comprising more than two proposals, and shows only the most prevalent code for categories in which multiple codes were combined. The complete list of categories for each referent that was used to calculate the agreement rates can be found in Table 3 in Appendix A. Examples of proposals representing the largest categories are available in Appendix B. (For the sake of brevity, we do not include examples of proposals involving GUI elements, as those were not intended by participants as command sketches, which are the focus of this study.)

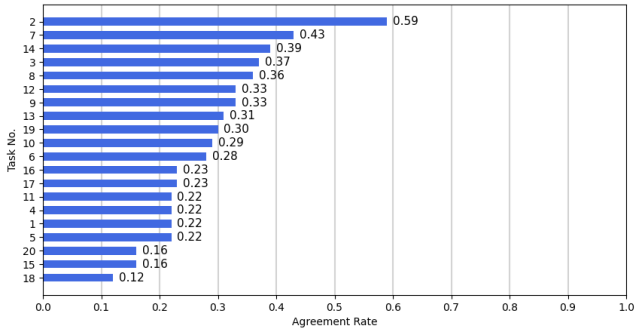



**Figure 3: Agreement rates for all tasks**

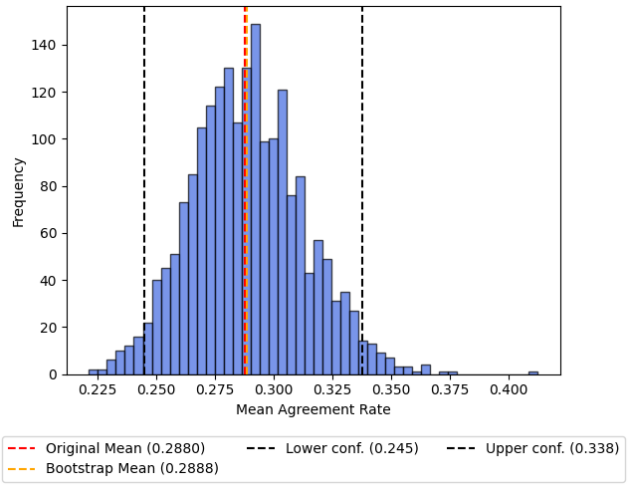



**Figure 4: Mean agreement rates across all tasks**

**Table 2: Agreement rates of tasks and breakdowns of categories with more than two proposals**

| Task | Title | **AR** | A | $P_1$ | $P_2$ | $P_3$ | Proposals | Example |
|---|---|---|---|---|---|---|---|---|
| 1 | Change value | 0.22 | IMP | TXT | SEL | | 14 | Figure 20 |
| | | | IMP | TXT | LOC.P1 | | 11 | Figure 21 |
| | | | IMP | GUI | SEL | | 5 | |
| 2 | Delete value | 0.59 | IMP | SEL | | | 27 | Figure 22 |
| | | | GUI | SEL | | | 7 | |
| 3 | Delete values | 0.37 | IMP | SEL | (range) | | 21 | Figure 23 |
| | | | GUI | SEL | | | 8 | |
| | | | IMP | SEL | (cells) | | 4 | |
| 4 | Add two values | 0.22 | TXT | TXT | TXT | LOC.A | 13 | Figure 24 |
| | | | SYM | SEL | SEL | LOC.A | 10 | Figure 25 |
| | | | GUI | SEL | SEL | SEL | 6 | |
| | | | TXT | SEL | SEL | LOC.A | 4 | |
| | | | SYM | LOC.A | LOC.A | SYM | 3 | |
| 5 | Insert column | 0.22 | GUI | SEL | | | 12 | |
| | | | IMP | SEL | | | 9 | Figure 26 |
| | | | SYM | LOC.A | | | 9 | Figure 27 |
| | | | TXT | SEL | | | 3 | |
| 6 | Remove column | 0.28 | IMP | SEL | (header) | | 16 | Figure 28 |
| | | | IMP | SEL | (range) | | 12 | Figure 29 |
| | | | GUI | SEL | | | 9 | |
| 7 | Move cells | 0.43 | CON | SEL | CON.END | | 21 | Figure 30 |
| | | | GUI | SEL | SEL | | 8 | |
| 8 | Add multiple values | 0.36 | SYM | SEL | LOC.A | | 20 | Figure 31 |
| | | | GUI | SEL | SEL | | 8 | |
| | | | TXT | TXT | LOC.A | | 5 | |
| 9 | Format values | 0.33 | IMP | SEL | GUI | | 16 | |
| | | | IMP | SEL | SYM | | 12 | Figure 32 |
| | | | TXT | SEL | SYM | | 5 | |
| 10 | Frame cell | 0.29 | GUI | SEL | | | 15 | |
| | | | IMP | SEL | | | 12 | Figure 33 |
| | | | SYM | SEL | | | 4 | |
| | | | TXT | SEL | | | 4 | |
| 11 | Transfer formatting | 0.22 | GUI | SEL | SEL | | 12 | |
| | | | CON+TXT | SEL | SEL | | 7 | Figure 34 |
| | | | TXT | SEL | SEL | | 5 | |
| | | | IMP | SEL | SEL | | 3 | |
| 12 | Create pie chart | 0.33 | GUI | SEL | | | 15 | |
| | | | EXA | SEL | | | 14 | Figure 35 |
| | | | TXT | SEL | | | 5 | |
| 13 | Create bar chart | 0.31 | GUI | SEL | | | 15 | |
| | | | EXA | SEL | | | 14 | Figure 36 |
| | | | TXT | SEL | | | 7 | |
| 14 | Continue series | 0.39 | IMP | SEL | SEL | | 22 | Figure 37 |
| | | | GUI | SEL | SEL | | 5 | |
| 15 | Transpose values | 0.16 | GUI | SEL | SEL | | 12 | |
| | | | CON | SEL | SEL | | 6 | Figure 38 |
| | | | TXT | SEL | CON.END | | 5 | Figure 39 |
| | | | SYM | SEL | SEL | | 4 | Figure 40 |
| 16 | Sort data | 0.23 | GUI | SEL | GUI | SEL | 16 | |
| | | | IMP | IMP | TXT | SEL | 5 | Figure 41 |
| | | | IMP | SEL | TXT | LOC.P2 | 5 | Figure 42 |
| 17 | Calculate multiple sums | 0.23 | SYM | SEL | SEL | (column) | 14 | Figure 43 |
| | | | GUI | SEL | SEL | | 9 | |
| | | | TXT | TXT | SEL | | 5 | |
| | | | SYM | SEL | LOC.A | | 4 | |
| | | | SYM | SEL | SEL | (table) | 3 | |
| 18 | Construct formula | 0.12 | GUI | SEL | SEL | SEL | 9 | |
| | | | TXT | TXT | TXT | SEL | 7 | Figure 44 |
| | | | SYM | SEL | SEL | SEL | 5 | Figure 45 |
| 19 | Conditional formatting | 0.30 | GUI | SEL | GUI | GUI | 14 | |
| | | | IMP | SEL | TXT | TXT | 11 | Figure 46 |
| | | | TXT | SEL | TXT | TXT | 10 | |
| 20 | Multiply values | 0.16 | SYM | SEL | SEL | LOC.A | 8 | Figure 47 |
| | | | GUI | SEL | SEL | SEL | 8 | |
| | | | TXT | TXT | TXT | LOC.A | 7 | Figure 48 |
| | | | TXT | TXT | SEL | LOC.A | 6 | Figure 49 |
| | | | CON+TXT | SEL | SEL | CON.END | 4 | |

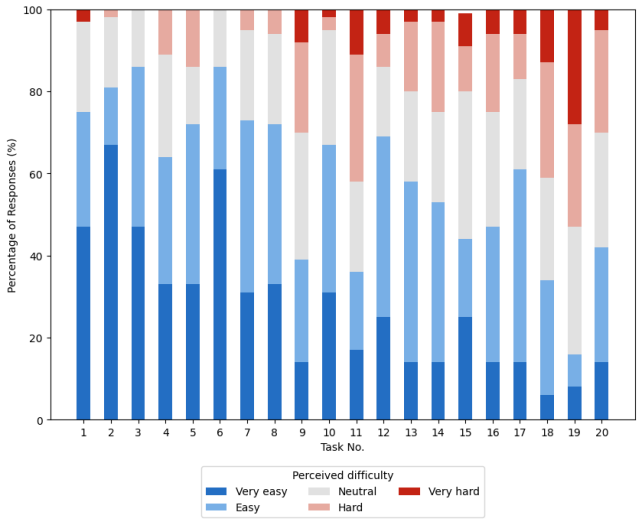



**Figure 5: Participants' perceived difficulty of referents**

In addition to capturing the sketches drawn by participants, we also asked them about the perceived difficulty of each referent and their satisfaction with their proposal on a five-point scale ranging from very negative to very positive ratings. Figure 5 shows the distribution of referents' perceived difficulty, while Figure 6 shows the distribution of participants' satisfaction with their proposals.

## 5 Discussion

To look at agreement not only from the the perspective of tasks but of participants, we broke down the sizes of the categories that participants' proposals were classified into (Figure 7). This chart reveals that seven out of 36 participants (19%) tended to produce rather individual proposals for at least half the referents, each of

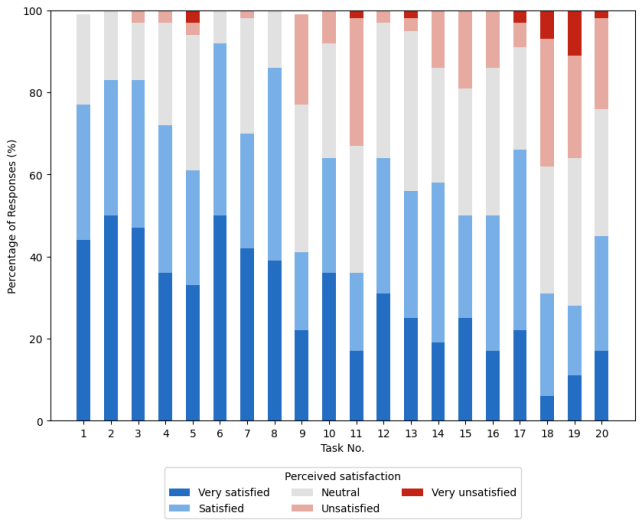



**Figure 6: Participants' perceived satisfaction with proposals**

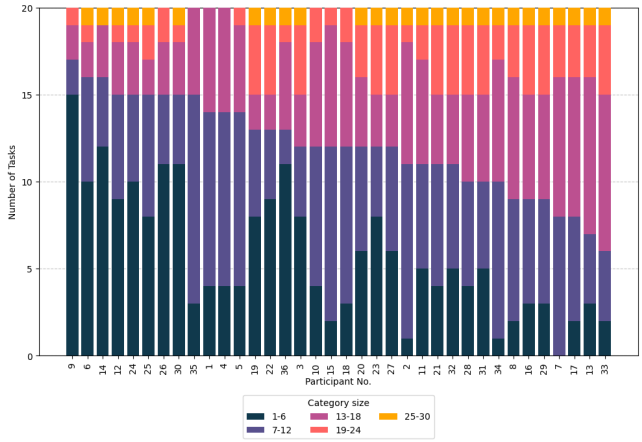



**Figure 7: Breakdown of size of categories that participants' proposals were assigned to**

which were in agreement with less than six other participants' proposals. On the other hand, 58% of participants produced proposals that were in agreement with at least 33% of proposals for at least 40% of referents.

The breakdown of participants' perceived difficulty of referents (Figure 5) and perceived satisfaction with their proposals (Figure 6) shows that perceived difficulty tends to grow with increasing task number, indicating that our assignment of tasks to the difficulty groups W, A, B, and C was largely accurate. Perceived satisfaction tends to be higher than perceived difficulty, suggesting that participants generally did not feel overwhelmed by the tasks.

Examining the participants' sketches in the light of our research questions, we can identify a number of patterns in how users express their command intentions. We next discuss these (Section 5.1) and also comment on more general observations we made across tasks (Section 5.2), as well as anecdotal observations that might seem like individual participants' idiosyncrasies, but highlight interesting mental models or approaches to sketching that are worthy of further investigation (Section 5.3). We finally discuss legacy bias (Section 5.4) as well as limitations and threats to validity (Section 5.5).

### 5.1 RQ1: Common Sketched Command Expressions

Although the agreement rates we observed can be considered medium to high according to Vatavu and Wobbrock [43], we observed clear diversity in the visual syntax of our participants' proposals. While their general idea may often be the same, their visual execution often differs, as can be seen e.g., in the various styles for selecting cells (Section 5.2). Deriving a set of user-defined interactions, as is common in the elicitation of gestures, seems therefore infeasible for sketched commands. We instead need to abstract from the individual ink strokes, and consider the shapes they form, their relations to each other, and the meaning that users assign to them in the context of the underlying spreadsheet substrate.

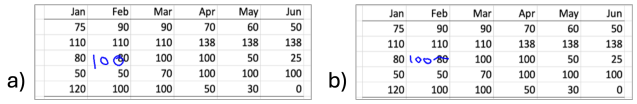



**Figure 8: Examples of common proposals for Task 1**

We do this first in the following subsections by identifying patterns that answer our RQ1.1–1.5 about common expressions of different types of commands, and then examine the more general implications for RQ2 in Section 5.2.

*5.1.1 RQ1.1: Editing data.* The proposals for Task 1 (Enter value) largely fell into one of two categories: Participants either overwrote the previous cell contents with the new value (Figure 8a), or struck through the old contents and wrote the new value beside the cell (Figure 8b). This is in line with how one would correct a value on paper, and how most participants also expressed local edits in an IDE [30]. Surprisingly, however, a number of participants came up with solutions that seemed unnecessarily complex, e.g., invoking classic UI elements or drawing explicit instructions to express the replacement of a value, which yielded an unexpectedly low AR of just 0.22. This may have been due to participants still warming up to the idea of sketching right on the spreadsheet.

Agreement within Tasks 2 and 3 (Delete value[s]) was higher, with most participants simply striking through the content to be deleted (Figure 9a), but even here, several participants wanted to invoke a GUI to complete the task. One participant proposed a sketch resembling a touch gesture that "pulled the old value out of the sheet". For deleting a range of values (Task 3), some participants just crossed out the whole range (Figure 9b), while others struck through every cell individually (Figure 9c).

In contrast to many more trivial editing operations, Task 14 (Continue series) has no direct paper equivalent. Despite this, it yielded a quite high AR of 0.39, with many participants employing arrows to convey the direction and extent of the desired continuation of the series (Figure 9d). We discuss repetition patterns in more detail in Section 5.2.

*5.1.2 RQ1.2: Structural changes.* We asked users to perform two kinds of structural changes: Tasks 5 and 6 dealt with inserting

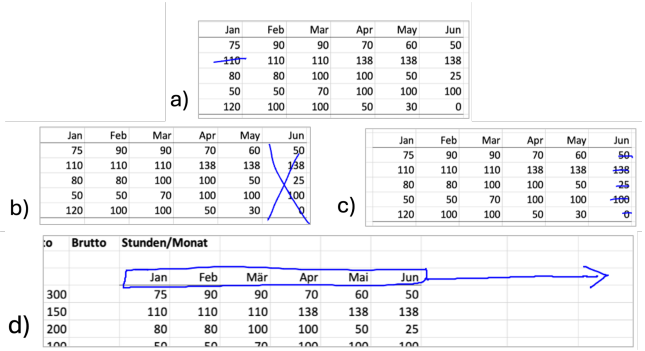



**Figure 9: Examples of common proposals for Task 2 (a), Task 3 (b, c), and Task 14 (d)**

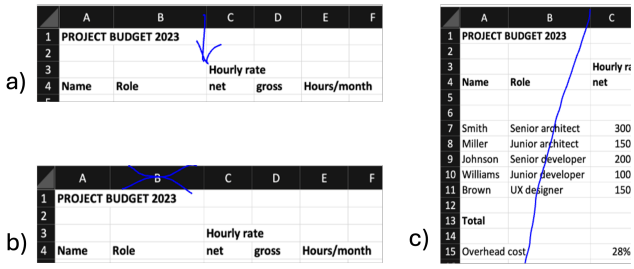



**Figure 10: Examples of common proposals for Task 5 (a) and Task 6 (b, c)**

columns into and deleting them from the grid. These commands were often expressed like one might mark up a printed table, i.e., by using insertion markers for Task 5 (Figure 10a) and either crossing out a column's header (Figure 10b) or striking through all its cells (Figure 10c) for Task 6 (Remove column). Innovative individual solutions for Task 5 (Insert column) included expressions for moving an existing empty column to the insertion location or drawing an example of an empty column.

In contrast to these manipulations of the grid itself, Tasks 7, 15, and 16 involved rearrangements of content *within* the grid, i.e., changes to the data's tabular organization. The proposals for Task 7 (Move cells) exhibited a quite high AR of 0.43 and were largely in line with the arrow notation favored by IDE users for the same purpose [30] (Figure 11a). Participants seemed to struggle considerably more with Task 15 and Task 16, though: For both tasks, proposals involving GUI usage represented the largest categories (though not the majority). For Task 15 (Transpose values), the proposals appear quite diverse when focusing only on the coding, but the large majority of sketch-based solutions in fact follow the same basic pattern of selecting the (vertically oriented) cell range to be transposed, and relating it to the (horizontally oriented) destination range by means of a connection, text, or symbol (Figure 11b). For Task 16 (Sort data), the sketch-based proposals are more fragmented in both visual and coding terms. Some of this diversity is certainly due to the fact that with an increasing number of parameters, there is more opportunity

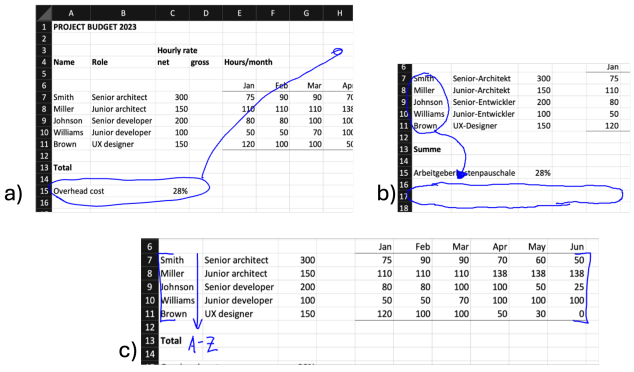



**Figure 11: Examples of common proposals for Task 7 (a), Task 15 (b), and Task 16 (c)**

for slightly divergent solutions. Looking at the proposals through a less precise lens than the individual codes though, the three main strategies to solve this task were reliance on GUI elements, selection of the affected cells and writing a textual command, and (in the minority of proposals) a comparatively simple symbolic expression of the sort operation and direction (Figure 11c).

This indicates that although an elegant and visually lightweight sketch expression for the sorting command could be conceived, many participants shied away from the operation's perceived complexity and resorted to executing it via a GUI or describing it textually. We hypothesize that the leaner visual solution would likely be favored by users over the more complex GUI- and text-based solutions once it was presented to them, and that it would be adoptable with low learning effort. A follow-up study that presents a curated set of sketches to participants and asks for their rating (as Samuelsson and Book [31] conducted for sketching on IDEs) would be an interesting subject of further research.

*5.1.3 RQ1.3: Formatting.* Our study covered a quite diverse set of formatting tasks, ranging from rather trivial operations such as framing a cell (Task 10) to more complex formats such as currencies (Task 9), and operations without a static target format such as transferring an existing format to other cells (Task 11) and defining a conditional formatting rule (Task 19). Interestingly, for each of these tasks, about 40% of participants produced proposals relying on GUI elements rather than resorting to sketching.

To our surprise, this was even the case for the almost trivially sketchable Task 10 of drawing a single frame around a cell. Our assumption is that this is indicative of heavy legacy bias – users likely remember border-drawing as an operation that involves complex dialogs in classic mouse-driven UIs and might therefore consider the task too complex to sketch. A related explanation might be that users are so familiar with this feature's potential complexity that they thought too much about extensions of the task beyond the referent, and did not deem sketched commands capable of expressing all that variety, even when it was not asked for here.

When designing this task, we were mostly wondering if and how participants would disambiguate between selecting cells and framing cells, when the most obvious expression for both would be sketching an enclosure around them. Most participants do not seem to have bothered to make this distinction (which is a fair approach, given that we instructed them to assume that sketches

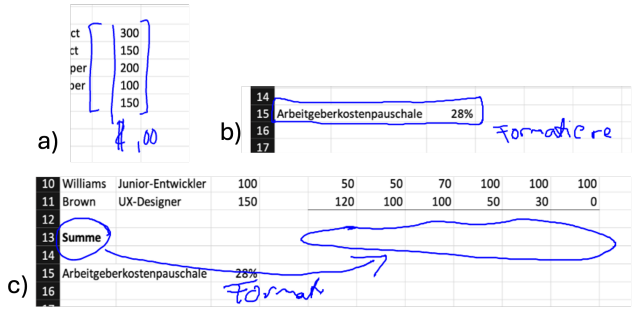



**Figure 12: Examples of common proposals for Task 9 (a), Task 10 (b), and Task 11 (c)**

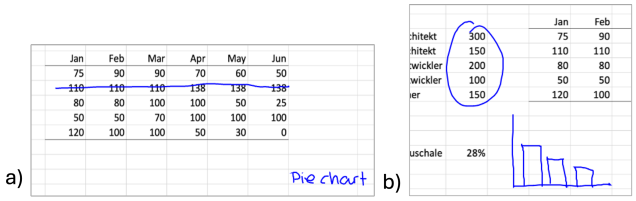



**Figure 13: Examples of common proposals for Task 12 (a) and Task 13 (b)**

would automatically be interpreted correctly by the hypothetical tool). Those users who did take steps to distinguish framing from selection did so by adding a command text or symbol (Figure 12b).

In Task 9 (Currency format), most participants who pursued a sketch- rather than a GUI-based solution simply expressed the format by writing a currency symbol – some with the assumption that this would imply the common two-decimal format, while others spelled that format out in examples such as "$,00" (Figure 12a). The fact that 44% of participants nevertheless produced a GUI-based proposal is an indication of the power of legacy bias that prevents participants from conceiving much simpler command expressions than their familiar interaction modalities provide. This observation was repeated in Task 11 (Transfer formatting), where one third of participants explained how they would use somewhat awkward and platform-atypical GUI menus to accomplish what the sketch-based solutions simply expressed through two area selections with a command word in between (Figure 12c).[5]

In line with our expectations, Task 19 (Conditional formatting) was rated by participants as the most difficult task of all, presumably due to its complexity and lack of obvious symbolic representations. Proposals were split between 39% GUI users and 61% sketch users, who employed more or less verbose textual formulae to describe the formatting condition and outcome (with the most concise idea, and the authors' favorite, employing the ternary conditional operator available in many programming languages (Figure 1d)).

*5.1.4 RQ1.4: Visualization.* We included the two very similar tasks of creating a pie chart (Task 12) and a bar chart (Task 13) from a given range of data to examine how consistent participants would be in their proposals (since they would not see these tasks back to back but interspersed with others, due to the randomized sequence within the task groups). In both cases, the proposals were almost evenly split between participants expecting to use a GUI for this purpose and participants sketching an example of the chart (Figure 13b), plus a few participants who simply wrote a textual command to create the chart (Figure 13a).

While the textual command might initially appear to underutilize the expressive power that the sketch modality offers (by foregoing the ability to sketch the example chart in the desired style, such as in 2D or 3D, horizontal or vertical), we speculate that some participants who wrote a simple text command might have understood this as a first step in an iterative-interactive process of

[5] In Task 11, we excluded four proposals because the participants solved a different task than prompted: Seeing that the source cells were in bold face, they sketched a command to format the destination cells in bold face as well, rather than sketching a generic command to transfer any kind of formatting.

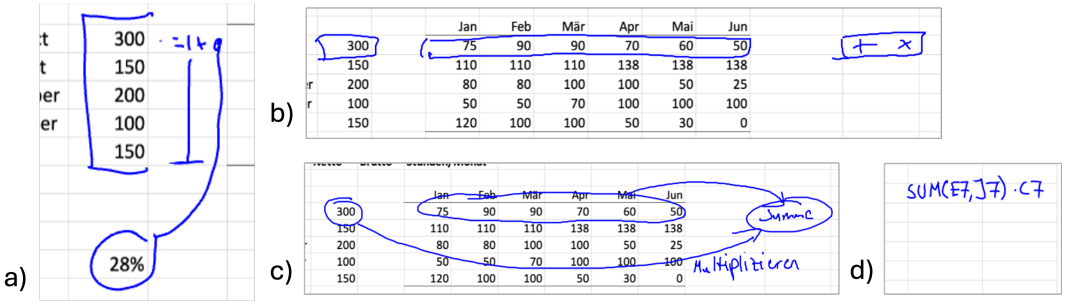



**Figure 14: Examples of common proposals for Tasks 18 (a) and Task 20 (b, c, d)**

visually refining a diagram proposed by the tool – by e.g., adding perspective to it, adding or striking out labels, etc. Such iterative refinement of charts towards the look desired by the user currently requires tedious navigation of menus, but could be achieved quite intuitively through sketching on top of the intermediate visualizations. Exploring effective ways for implementing this process will be an interesting area of future research.

*5.1.5 RQ1.5: Calculation.* Calculations are the core feature of any spreadsheet application and undoubtedly the most powerful and complex one. Yet, despite the central importance of this feature and the obvious high mental load involved in conceiving, writing, debugging, and maintaining formulae in spreadsheets, the UI for interacting with them in common software tools is a tiny single input line that is already challenging to handle in a desktop environment, and almost unusable for non-trivial formulae on a touch-only device such as a tablet.

We hypothesize that expressing the structure or outcome of mathematical operations and other formulae visually, instead of typing them out on a single line, could be a much more user-friendly way of accessing this core feature (while likely coming with usability challenges of its own, such as incorporating cells into a sketch that are currently off-screen). Sketching could play an interesting role among the approaches being discussed as alternatives to textual formula construction (e.g., using natural language [35]), and certainly offers considerable opportunities for further research.

As a first step that likely only scratches the surface of what might be possible here, we included five calculation tasks in our study: Calculating the sum of two values (Task 4), the sum of multiple values (Task 8), multiple sums of multiple values (Task 17), multiple products of a value and a percentage (Task 18), and the combination of a sum and a product (Task 20).

Most proposals for Task 4 (Add two values) and Task 8 (Add multiple values) involved either selections of the involved summands and an operator symbol (Figure 15a) or writing out the addition formula (Figure 15b). Interestingly, several proposals leveraged conventions about the table structure to imply command aspects, such as foregoing an explicit selection of the summands, but assuming that all values in a column should be added when writing a Σ symbol underneath them (Figure 15c). Task 17 is an interesting extension of Task 8 because it performs the same addition multiple times in adjacent columns of a table. Many participants expressed this by simply adding a line or arrow across the cells in which the operation should be repeated (Figure 15d).

Task 18 asked participants to increase each value in a set by a given percentage. This task was perceived to be the second-most difficult, and it yielded the lowest AR of all tasks (0.12), as participants apparently struggled to express the colloquial usage of "adding a percentage" that obscures its precise mathematical definition ($value * (1 + percentage)$), as well as the repetition of the operation across multiple rows. The majority of participants (75%) chose a more or less complete textual or symbolic representation of the formula, and represented the repetition across rows graphically (Figure 14a). A quarter of the participants however wanted to use some form of GUI for this task, even though menus, buttons and pop-ups are neither common nor particularly useful means of constructing arithmetic formulae. We again explain this with legacy bias – faced with an apparently complex problem, participants seemed to trust a GUI to be the most suitable way to solve it, even though most of the GUI-based proposals for this task tended to be impractical.

Finally, the combination of addition and multiplication in Task 20 was solved by participants in four distinct ways: 45% chose to express the operation by writing down some variation of the formula (Figure 14d), inspired either by arithmetic syntax or known spreadsheet functions. 22% wanted to employ some form of GUI, and 11% connected selections of the relevant values and operators

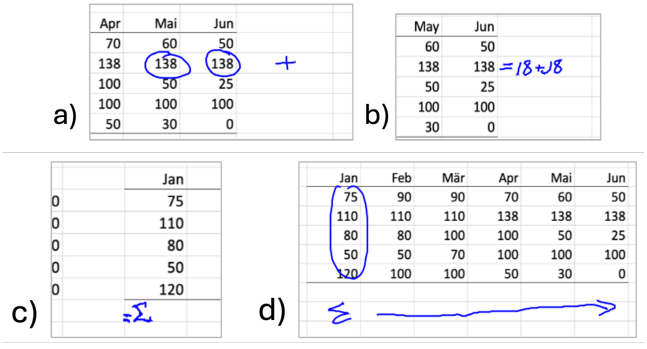



**Figure 15: Examples of common proposals for Task 4 (a, b), Task 8 (c), and Task 17 (d)**

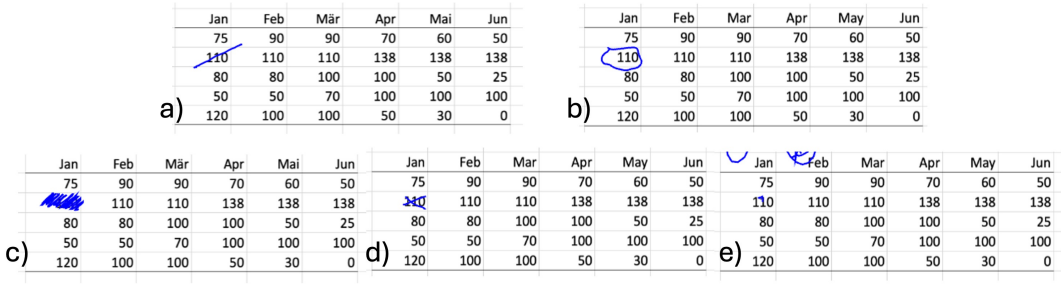



**Figure 16: Examples of selection styles, all taken from Task 2: a) strike (can also be an underline), b) enclosure (typically by encircling), c) squiggly line, d) X mark, and e) tap (barely visible dot on value 110 in first column)**

with lines and arrows (Figure 14c). Another 22% also selected the relevant value ranges, but placed the addition and multiplication operators in a seemingly unrelated manner when looking merely at the static final sketch image (Figure 14b). A review of the screen recordings however revealed that these participants ascribed meaning to the temporal sequence in which they drew the selections and the operators, expecting the values to be added and multiplied in the order that they sketched the respective elements.

Understanding that users might employ this strategy is important for sketch recognizers and interpreters, as they must not only rely on the finished sketch produced by a user, but need to take the temporal genesis of the sketch into account. This is not just a challenge, but also an opportunity, as it allows the tool to provide visual feedback to the user as the sketch evolves (for example by calculating intermediate results), thereby showing the user whether they are on the right track towards achieving their desired outcome.

## 5.2 RQ2: General Sketch Interpretation Considerations

In examining the sketched command expressions that participants proposed for our referents, we observed a number of patterns that were independent of particular tasks. In the following subsections, we discuss the strategies that participants (likely subconsciously) pursued with these, and their implications for the design of sketch interpretation engines.

*5.2.1 Interpretation of arrows.* We found that participants used arrows for a variety of purposes, such as to indicate repetition, movement, and conceptual relations, going beyond the findings of Samuelsson and Book [31], whose participants used arrows predominantly to convey movement of content in an IDE. Besides this variety in *purpose*, we also observed a large variety in arrows' *meaning*, namely:

- Arrows whose start, end, and path did not carry meaning, but which were used symbolically or as operators (SYM), e.g., to indicate a consequence.
- Arrows whose start- and endpoint mattered, but not their path. These typically indicated connections (CON) such as the origin and destination of a movement, relationship, or information flow.
- Arrows whose start- and endpoint were byproducts of their path, which primarily mattered in order to select (SEL) the involved cells.
- Arrows that were redundant or did not carry any relevant meaning at all, but were apparently only drawn by participants to make the sketch look more complete or clear to themselves. These were excluded from our coding, which was focused on identifying elements that carried the necessary information to interpret a command.

This variety in visual appearance, conveyed information, and larger purpose makes arrows an extremely powerful syntax element of sketched command expressions, but also a potentially ambiguous one, so sketch interpreters will require contextual clues to interpret arrows correctly.

*5.2.2 Selection styles.* Participants employed a diverse array of drawing styles when selecting single or multiple cells and values. The most common methods included straight lines, such as underlining or striking through content, and enclosing selections by encircling cells. Lines were sometimes embellished with arrows, even when used solely for selection purposes, without the direction indicated by the arrowhead conveying relevant meaning. Some participants opted for alternative approaches, such as using squiggly lines or drawing large X marks over cells. Tapping directly on individual cells was also a frequently observed selection method, especially when multiple cells should be combined in an arithmetic operation. Figure 16 shows the most common selection methods for a single cell, although in many tasks, participants selected multiple cells by simply extending their chosen variant (see Figure 17 for some enclosures spanning multiple cells). Participants were relatively consistent with their individual selection style, but we found some deviations in the data that hint at opportunities for further analysis.

*5.2.3 Repetition.* A common feature of spreadsheets is to continue sequences, i.e., to infer the content of the next cells automatically. This can be used to, e.g., define a series of months, a range of values, or copy a function to multiple cells. Our participants employed different approaches to express such commands, as can be seen in Figure 17: repeating the same sketch several times (b); selecting the area of repetition via an arrow, enclosure, or line (a, c); drawing brackets (e) or dedicated symbols delineating the repetition (d); or writing explicit commands like "complete series" or "continue" (f),

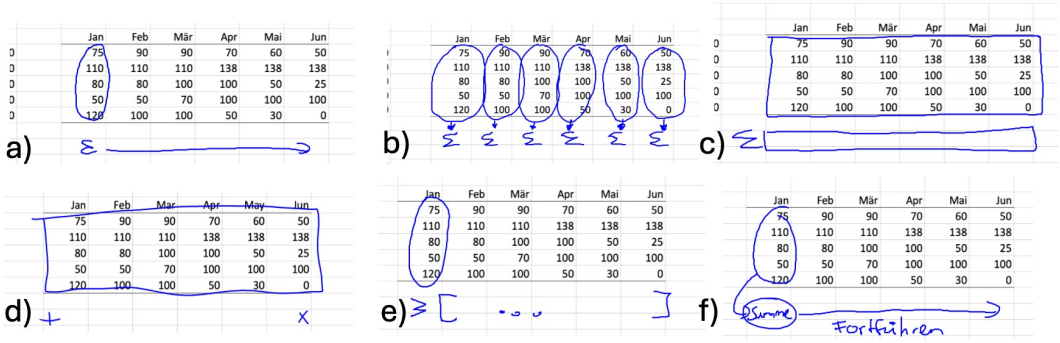



**Figure 17: Examples of repetitions, all taken from Task 17 (handwriting in (f) translates to "sum" and "continue")**

also in combination with an arrow. Another notable observation is that some participants selected all the operands used in every operation (Figure 17c, d), while others selected only the operands of a single operation and then indicated the repetition only for the result (Figure 17a, e, f), so an interpretation engine would have to infer the relevant input data even if it is not selected explicitly.

*5.2.4 Examples and references.* In several tasks, participants used examples to express commands. Examples could either be sketched expressions (e.g., drawing a stylized pie chart or defining a formatting for a currency), or references to other cells. Some participants also defined content references by, e.g., enclosing an area, labeling it with a name, and then referencing this elsewhere as a parameter. Figure 18 shows an example from Task 20: The participant labels several values with encircled numbers that are used again in cell K7 to multiply the referenced values. The participant also uses the numeral 1 as an anchor indicating which values the summation operator should apply to.

Participants employed a wide range of syntactical variants for declaring and referring to examples. While such constructs enable very efficient command expressions (especially for expert users), implementing an appropriate interpretation engine requires additional analysis of the underlying spreadsheet, in order to understand correctly how the referenced content should be interpreted in its new context (e.g., as a value, a template, in terms of its formatting, or in other ways).

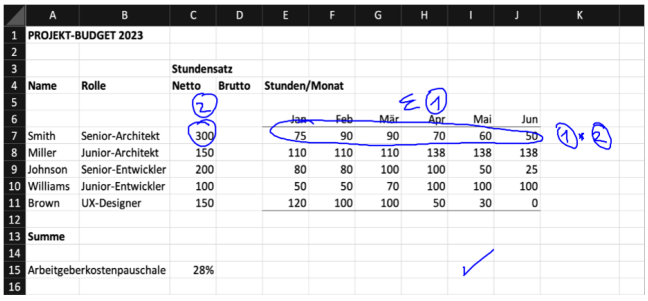



**Figure 18: Example proposal for Task 20 illustrating the use of references to construct an arithmetic operation**

*5.2.5 Implied command aspects.* Our participants frequently did not explicitly express all aspects of a command, but left the action or some parameter without a sketched representation. In most cases, that information was however not missing from the sketch expression, but could be inferred from one or more of the following contextual aspects:

- Certain *explicit sketch elements* can carry enough meaning to cover several command aspects. For example, the expression "A–Z" in Figure 1c denotes both an action ("sort") and one of its parameters ("ascending alphabetical order").
- Certain *visual styles* of sketch elements carry extended meaning. For example, drawing an X or a squiggly line over cells (Figure 16c, d) has the strong connotation of deleting that content, i.e., describing both the "delete" action and its range.
- The *substrate* that users are sketching on (in our study, the spreadsheet) is a particularly rich source of context: Some participants leveraged conventions about table layout (such as the common practice that the sum of the values in a column is usually placed underneath it) to imply the range of an addition operation that was only designated by a command symbol (Figure 15c). We conjecture that many more complex table operations could leverage such implications as well (or may even only be practically expressible because of them).
- The *temporal sequence* of sketches can serve as a useful construct to designate relations between command aspects (such as the operators and operands in Figure 14b) without the need to sketch explicit connections between them.

In our session recordings, it seemed that participants rarely made a deliberate decision *not* to sketch a certain command aspect explicitly – they rather left those aspects out because the sketch apparently made intuitive sense to them without them. In contrast, a minority of participants tended to "over-engineer" their sketches by adding redundant elements or explicitly stating that they included particular elements to clarify command aspects for the recognition logic. Participant 23, for example, finished every sketch by drawing a checkmark as an indication that they were done (see Figure 18 for an example), much like pressing "Enter" after typing a command-line argument.

Implying parts of a command seems to be a powerful strategy for making sketched commands more intuitive, efficient and learnable.

However, implication poses challenges for sketch recognition and interpretation components in future sketch-enabled spreadsheet applications because they need to consider a number of very different contextual cues (ranging from the spatial and temporal coordinates of ink strokes to the layout and possibly domain-specific semantics of the underlying spreadsheet) in order to derive the implied information, resolve potential ambiguities, and correctly interpret the sketch.

## 5.3 Unorthodox Uses of Sketching

While most participants followed our instruction to express commands through sketching (aside from occasionally falling back to GUI-based interaction), two participants consistently envisioned unique interaction modes that deviated from the others.

Participant 9 refrained from sketching graphical elements for most referents, and instead relied on handwriting formulae into an imaginary input field above the spreadsheet, mixing mathematical and programming notation (Figure 19a). Participant 35 in contrast used another recurring scheme for each referent: While they typically used a sketched enclosure to select the main subject of the command, they expressed the action by drawing a box resembling a terminal or chat box on the screen and filled it with a natural-language command (Figure 19b).

It seemed that both wanted to "talk" directly to the application to perform some action, albeit in different ways: While Participant 9's style seemed to be influenced by their apparent technical background, Participant 35's approach reflects the idea of having an interactive assistant, which would be feasible today given the advances in large language models (LLMs) and handwriting recognition. The LLM would require capabilities to process the context of the rather unspecific textual commands, i.e., spreadsheet and sketches, but this should be possible with multi-modal LLMs or effective pre-processing of the sketch to identify the relevant data.

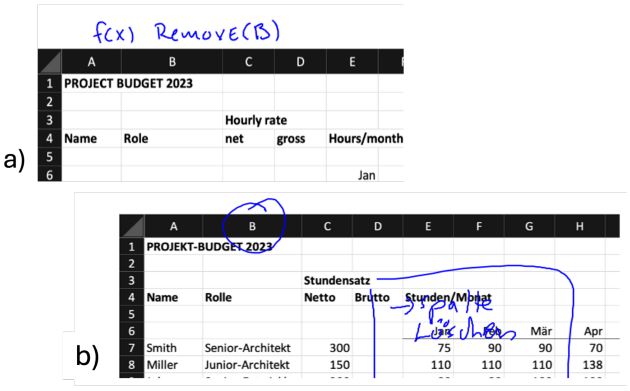



**Figure 19: Examples from Task 6 for two participants consistently deviating from the instructions: Participant 9 (left) relying on written formulae, and Participant 35 (right) using sketching for selection and an envisioned text box to "chat" with the application (handwriting in (b) translates to "delete column")**

Uttering the commands that Participant 35 handwrote into their envisioned chat box as voice commands (similar to, e.g., GridBook [35]) would offer another interesting potential for direct interaction using multiple modalities.

## 5.4 Legacy Bias

While elicitation studies as a form of empirical work are prone to several limitations and threats to validity (see Section 5.5 for details), an effect known as *legacy bias* stands out in our observations. There is a tendency for participants in elicitation studies to subconsciously rely on familiar patterns from previous tasks and experiences when faced with new tasks, especially if they refer to already known applications. Even Wobbrock et al. [48], which can be considered the progenitors of elicitation studies, reported that participants fell back on elements known from classic desktop computing, although they had specifically designed their study not to resemble well-known computing environments. With this in mind, spreadsheets must be extremely prone to legacy bias as they have been around for decades (the first applications can be traced back to the 1960s) and are a common application in many workplaces today.

Our data reveals the presence of considerable legacy bias, as many participants fell back on GUI components such as buttons and menus even when other participants came up with quite elegant sketched commands for certain tasks. While GUI usage was lowest for most tasks in the groups we considered easiest (Groups W and A), it turned out that more participants relied on GUI elements for the tasks in Group B than Group C, despite Group C being considered more difficult.

Notably though, the tasks with the highest ratio of GUI-based proposals (39% and more) are those that involve complex GUIs in common spreadsheet applications, such as defining number formats (Task 9) and conditional formatting rules (Task 19), creating charts (Tasks 12 and 13) and sorting data (Task 16), or even just framing a cell (Task 10). In contrast, the ratio of GUI-based proposals for arithmetical tasks was 26% or less, even for the most complex calculations such as Tasks 17 and 18 – most likely because participants are not used to solving these tasks using GUI menus and buttons.

Therefore, while we had expected GUI reliance to be largely influenced by task complexity, we found legacy bias to be a stronger factor among our participants. It will be interesting to study if this bias can be overcome by presenting users with attractive (i.e, easily learnable and efficient) sketch-based alternatives to the GUI-based commands they are used to.

## 5.5 Limitations and Threats to Validity

Like many empirical studies, our elicitation study also is subject to some limitations. Our participants were aged 18–40 and therefore on the younger side of the age spectrum; a common issue in elicitation studies [45]. As differences in elicited gestures between age groups have been observed [12], it remains unclear how this affects our results.

Instead of presenting participants with a referent showing the state before and after their interaction, as is common for gesture elicitation studies, we opted to use a textual prompt referring to a static image of the same simple spreadsheet throughout the study. This was done as we felt the visual impact of showing two spreadsheets,

and letting participants understand the differences between them, would have been overwhelming. While this meant that participants had to map the textual prompt to the structure of the spreadsheet, and especially identify the affected cells, we observed only very few cases where participants clearly misunderstood the prompt and solved a different task than intended. Furthermore, showing two large spreadsheets (before and after) would reduce the screen real estate available for sketching massively, which would eliminate the main benefit of sketching as an interaction modality that does not require any screen real estate to be taken away from the presentation of the user's document.

In our study setup, the sketching area available to the participants measured about three quarters of a 12.4" tablet's screen. While we assume that this is one of the most common hardware environments in which participants would work with a stylus, sketching on spreadsheets is also conceivable in different environments covering a large range of form factors, e.g., on pen-equipped (potentially foldable) smartphones, on 2-in-1 laptops that can be folded like tablets, on large wall-mounted interactive displays, and potentially even in augmented or virtual reality environments. It is important to note that even without an on-screen keyboard encroaching on the available sketch space, the different sizes of these devices would likely influence the level of precision, detail, and complexity that users can achieve in their sketches. We hypothesize that many of the basic patterns we have discussed in the preceding sections are device-independent. We would therefore discourage the design of different sketching languages for different screen sizes. Rather, we envision a sketching language whose basic constructs are expressible on any screen size, but that allows for richer command expressions on larger screens. The design of such a visual command language for spreadsheets is a major goal of our ongoing research.

Regarding our study setup, it is also important to note that we used a static image of a sample spreadsheet as the substrate to sketch on, so our application was not interactive in the sense discussed in Sect. 2.1, as it did not provide immediate feedback beyond displaying the digital ink. This is an intrinsic limitation of elicitation studies, where the participants' responses cannot be anticipated. Our participants did not seem to be disturbed by this though – from the think-aloud recordings, it rather became clear that they imagined the application responding in certain ways, and sketched their interaction as it would proceed with a responsive application. This was especially apparent with participants who favored GUI-based solutions, and would often include several consecutive interaction steps in the same sketch of their imaginary GUI. While we did not see multi-step interactions expressed as explicitly among the participants who drew sketches rather than GUI elements, we assume they would have been able to make this leap of imagination as well if they had considered it necessary. In future usability studies of truly responsive prototypes of sketch-enabled applications, it will be interesting to contrast single- and multi-step sketch interactions.

While not a limitation per se, we lastly note that our participants were end users and thus laypeople with only little to no experience in interaction design. While this is the core idea of elicitation studies, it might also explain the extent of legacy bias we observed. However, different approaches to the design of e.g., gesture-based interaction have been proposed [50] – expert-led, user-led, or computationally-based – that we argue can be applied to sketch-based interaction as well. Elicitation studies fall into the user-led category, but interviews with expert designers might yield other ideas of how sketching can be used as an interaction modality.

## 6 Conclusion and Future Work

We presented results from an elicitation study that examines how users would express commands for manipulating a spreadsheet by sketching directly on the application's GUI. Our results indicate that while users can envision syntactically quite diverse ways of expressing the same command, the conceptual strategies and patterns they employed to convey meaning in a visual way were quite coherent. We highlight several such patterns, e.g., to convey selection, repetition, and strategies (such as the use of examples, references and implications) that are of central importance for the design of sketch recognition and interpretation components in future sketch-enabled user interfaces.

Our study highlighted several aspects that require further research on the way towards such interfaces:

Regarding the elicitation and definition of patterns (or even a visual language) for expressing commands through sketches, we have only explored a fraction of the functionality of spreadsheets or other applications. Considerable potential exists in examining how more complex operations (in particular, spreadsheet formulae) could be expressed through sketching, and how sketching might be used in multi-step interactions, such as iteratively refining the visual representations of data in charts.

One of the main design considerations for such a visual command language needs to be its ease of adoption and use, i.e., its learnability: Given how natural it is for humans to pick up a pen, we would want them to be able to sketch most commands without having to think about them – either because they can use common paper-based expressions (like scratching out a value to be deleted), or because they can combine or interpolate known command expressions ("if I can modify single cells like this, I can probably modify a whole cell range in the same way"), or because they can understand and remember a command at first glance when they look it up ("right, of course you'd draw an arrow to move what you've selected").

We speculate that especially for some classes of complex formulae, sketching a desired operation might be easier than remembering the relevant functions and constructing a syntactically correct formula from them. For more trivial operations, users might still favor sketching over other input modalities on devices without a classic mouse and keyboard (i.e., most mobile devices and wall-mounted displays), where using an on-screen keyboard would be too cumbersome as it typically obscures a large part of the content that the user is trying to work with. The convenience afforded by sketching in such environments that lack other precise input modalities might provide additional motivation for users to learn sketch commands.

To enable sketch-based interaction with software tools in practice, suitable sketch recognition and interpretation components are needed. Since the design of the command language will imply requirements for the interpretation technology, and the choice of interpretation technology might impose constraints on the properties of the visual language, these two endeavors need to proceed in close coordination. It will be interesting to examine whether it is possible and useful to create precisely defined sketching languages

that can be interpreted based on concrete heuristics, or whether the breadth of functionality and visual variety necessitates the development of interpretation approaches based on machine learning and artificial intelligence.

While existing techniques from gesture recognition (see, e.g., Magrofuoco et al. [18]) can be reused for shape detection, interpretation requires rich and varied contextual information involving not just the spatial and temporal properties of the digital ink strokes, but an understanding of the structure and semantics of the underlying data in the spreadsheet. Techniques from machine leaning and artificial intelligence should be helpful here; for example, Xu et al. [53] review recent advances in deep learning for free-hand sketching that can be adapted to understand spreadsheets as the sketching context.

Beyond this, sketching could be used in combination with other modalities, e.g., touch and sketching, or classic GUI elements and sketching, to combine their respective strengths. Achieving this in a natural way involves exploring the advantages and defining the realistic limitations of what can be accomplished with each type of interaction (and their combination). We therefore focused only on sketching in this study, leaving combinations to future work. Furthermore, combinations with modalities like text prompting or voice commands, enabling users to e.g., express an intended action vocally while expressing its parameters by sketching (for example, encircling a range of cells while saying "add these up", similar to Bolt's famous Put-That-There [1]), offer interesting possibilities. We speculate that sketching and speaking, humans' two most natural communication techniques, could complement each other ideally to enable the construction of highly effective direct-manipulation user interfaces, and we are planning to conduct further studies to explore this vision.

## Acknowledgments

Parts of this work were supported by the University of Iceland Research Fund (grant no. 92270) and the Icelandic Research Fund (grant no. 2410725). We would like to thank the anonymous participants for volunteering their time to contribute to this study.

## References

[1] Richard A. Bolt. 1980. "Put-that-there": Voice and Gesture at the Graphics Interface. *ACM SIGGRAPH Computer Graphics* 14, 3 (1980), 262–270. https://doi.org/10.1145/965105.807503

[2] Margaret M. Burnett and Herkimer J. Gottfried. 1998. Graphical definitions: expanding spreadsheet languages through direct manipulation and gestures. *ACM Trans. Comput.-Hum. Interact.* 5, 1 (1998), 1–33. https://doi.org/10.1145/274444.274445

[3] Bill Buxton. 2007. *Sketching User Experiences: Getting the Design Right and the Right Design.* Morgan Kaufmann.

[4] Vincent Cavez, Caroline Appert, and Emmanuel Pietriga. 2024. Spreadsheets on Interactive Surfaces: Breaking through the Grid with the Pen. *ACM Trans. Comput.-Hum. Interact.* 31, 2, Article 16 (2024), 33 pages. https://doi.org/10.1145/3630097

[5] George Chalhoub and Advait Sarkar. 2022. "It's Freedom to Put Things Where My Mind Wants": Understanding and Improving the User Experience of Structuring Data in Spreadsheets. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 585, 24 pages. https://doi.org/10.1145/3491102.3501833

[6] Mauro Cherubini, Gina Venolia, Rob DeLine, and Amy J. Ko. 2007. Let's Go to the Whiteboard: How and Why Software Developers Use Drawings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 557–566. https://doi.org/10.1145/1240624.1240714

[7] Ronald Chung, Petrut Mirica, and Beryl Plimmer. 2005. InkKit: a generic design tool for the tablet PC. In *Proceedings of the 6th ACM SIGCHI New Zealand Chapter's International Conference on Computer-Human Interaction: Making CHI Natural* (Auckland, New Zealand) *(CHINZ '05)*. Association for Computing Machinery, New York, NY, USA, 29–30. https://doi.org/10.1145/1073943.1073950

[8] Alan Dix and Layda Gongora. 2011. Externalisation and design. In *Procedings of the Second Conference on Creativity and Innovation in Design* (Eindhoven, Netherlands) *(DESIRE '11)*. Association for Computing Machinery, New York, NY, USA, 31–42. https://doi.org/10.1145/2079216.2079220

[9] Jonathan Fish and Stephen Scrivener. 1990. Amplifying the Mind's Eye: Sketching and Visual Cognition. *Leonardo* 23, 1 (1990), 117–126. http://www.jstor.org/stable/1578475

[10] Travis Gesslein, Verena Biener, Philipp Gagel, Daniel Schneider, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. 2020. Pen-based Interaction with Spreadsheets in Mobile Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 361–373. https://doi.org/10.1109/ISMAR50242.2020.00063

[11] Saul Greenberg, Sheelagh Carpendale, Nicolai Marquardt, and Bill Buxton. 2011. *Sketching User Experiences: The Workbook.* Morgan Kaufmann.

[12] Julia Hermann, Aysegül Dogangün, and Marc Hesenius. 2024. Age-Based Differences in Drone Control Gestures: An Exploratory Study. In *Proceedings of the 35th Australian Computer-Human Interaction Conference* (Wellington, New Zealand) *(OzCHI '23)*. Association for Computing Machinery, New York, NY, USA, 49–58. https://doi.org/10.1145/3638380.3638401

[13] Julia Hermann, Moritz Plückthun, Aysegül Dogangün, and Marc Hesenius. 2022. User-Defined Gesture and Voice Control in Human-Drone Interaction for Police Operations. In *Nordic Human-Computer Interaction Conference* (Aarhus, Denmark) *(NordiCHI '22)*. ACM, New York, NY, USA, Article 41. https://doi.org/10.1145/3546155.3546661

[14] Marc Hesenius, Markus Kleffmann, and Volker Gruhn. 2021. AugIR Meets GestureCards: A Digital Sketching Environment for Gesture-Based Applications. *Interacting with Computers* 33, 2 (2021), 134–154. https://doi.org/10.1093/iwcomp/iwab017

[15] Yea-Seul Kim, Nathalie Henry Riche, Bongshin Lee, Matthew Brehmer, Michel Pahud, Ken Hinckley, and Jessica Hullman. 2019. Inking Your Insights: Investigating Digital Externalization Behaviors During Data Analysis. In *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea) *(ISS '19)*. Association for Computing Machinery, New York, NY, USA, 255–267. https://doi.org/10.1145/3343055.3359714

[16] Markus Kleffmann, Marc Hesenius, and Volker Gruhn. 2015. Connecting UI and Business Processes in a Collaborative Sketching Environment. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Duisburg, Germany) *(EICS '15)*. ACM, New York, NY, USA, 200–205. https://doi.org/10.1145/2774225.2775076

[17] Seung Won Lee, Tae Hee Jo, Semin Jin, Jiin Choi, Kyungwon Yun, Sergio Bromberg, Seonghoon Ban, and Kyung Hoon Hyun. 2024. The Impact of Sketch-guided vs. Prompt-guided 3D Generative AIs on the Design Exploration Process. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 1057, 18 pages. https://doi.org/10.1145/3613904.3642218

[18] Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. 2021. Two-dimensional Stroke Gesture Recognition: A Survey. *ACM Comput. Surv.* 54, 7, Article 155 (2021), 36 pages. https://doi.org/10.1145/3465400

[19] Nicolas Mangano, Thomas D. LaToza, Marian Petre, and André van der Hoek. 2015. How Software Designers Interact with Sketches at the Whiteboard. *IEEE Transactions on Software Engineering* 41, 2 (2015), 135–156. https://doi.org/10.1109/TSE.2014.2362924

[20] Nicolai Marquardt. 2017. Sketching User Experiences: Hands-on Course of Sketching Techniques for HCI Research. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 1261–1263. https://doi.org/10.1145/3027063.3027107

[21] Soumik Mohian and Christoph Csallner. 2020. Doodle2App: Native App Code by Freehand UI Sketching. In *7th IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft)*.

[22] Soumik Mohian and Christoph Csallner. 2022. PSDoodle: searching for app screens via interactive sketching. In *Proceedings of the 9th IEEE/ACM International Conference on Mobile Software Engineering and Systems* (Pittsburgh, Pennsylvania) *(MOBILESoft '22)*. Association for Computing Machinery, New York, NY, USA, 84–88. https://doi.org/10.1145/3524613.3527807

[23] Meredith Ringel Morris, Andreea Danielescu, Steven Drucker, Danyel Fisher, Bongshin Lee, m. c. schraefel m. c. m. c., and Jacob O. Wobbrock. 2014. Reducing legacy bias in gesture elicitation studies. *Interactions* 21, 3 (2014), 40–45. https://doi.org/10.1145/2591689

[24] Brad Myers, Sun Young Park, Yoko Nakano, Greg Mueller, and Amy Ko. 2008. How designers design and program interactive behaviors. In *2008 IEEE Symposium on Visual Languages and Human-Centric Computing*. 177–184. https://doi.org/10.1109/VLHCC.2008.4639081

[25] Chris Parnin, Carsten Görg, and Spencer Rugaber. 2010. CodePad: interactive spaces for maintaining concentration in programming environments. In *Proceedings of the ACM 2010 Symposium on Software Visualization, Salt Lake City, UT, USA, October 25-26, 2010*, Alexandru Telea, Carsten Görg, and Steven P. Reiss (Eds.). ACM, 15–24. https://doi.org/10.1145/1879211.1879217

[26] Gary Perelman, Marcos Serrano, Christophe Bortolaso, Celia Picard, Mustapha Derras, and Emmanuel Dubois. 2019. Combining Tablets with Smartphones for Data Analytics. In *Human-Computer Interaction – INTERACT 2019*, David Lamas, Fernando Loizides, Lennart Nacke, Helen Petrie, Marco Winckler, and Panayiotis Zaphiris (Eds.). Springer International Publishing, Cham, 439–460.

[27] Gary Perelman, Marcos Serrano, Celia Picard, Christophe Bortolaso, Mustapha Derras, and Emmanuel Dubois. 2018. Cell Selection for Spreadsheets on Tablets: Stacking-Based Interaction. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3170427.3186546

[28] Felix Raab. 2016. Source Code Interaction on Touchscreens. Ph.D. Dissertation. University of Regensburg. https://nbnresolving.org/urn:nbn:de:bvb:355-epub-331075

[29] Hugo Romat, Nathalie Henry Riche, Ken Hinckley, Bongshin Lee, Caroline Appert, Emmanuel Pietriga, and Christopher Collins. 2019. ActiveInk: (Th)Inking with Data. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300272

[30] Sigurdur Gauti Samuelsson and Matthias Book. 2020. Eliciting Sketched Expressions of Command Intentions in an IDE. *Proc. ACM Hum.-Comput. Interact.* 4, ISS, Article 200 (2020), 25 pages. https://doi.org/10.1145/3427328

[31] Sigurdur Gauti Samuelsson and Matthias Book. 2023. Towards a Visual Language for Sketched Expression of Software IDE Commands. In *2023 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 115–123. https://doi.org/10.1109/VL-HCC57772.2023.00021

[32] Ugo Braga Sangiorgi, François Beuvens, and Jean Vanderdonckt. 2012. User interface design by collaborative sketching. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. Association for Computing Machinery, New York, NY, USA, 378–387. https://doi.org/10.1145/2317956.2318013

[33] C. Scaffidi, M. Shaw, and B. Myers. 2005. Estimating the numbers of end users and end user programmers. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*. 207–214. https://doi.org/10.1109/VLHCC.2005.34

[34] Ben Shneiderman. 1997. *Designing the user interface – Strategies for effective human-computer interaction, 3rd ed.* Addison-Wesley.

[35] Sruti Srinivasa Ragavan, Zhitao Hou, Yun Wang, Andrew D Gordon, Haidong Zhang, and Dongmei Zhang. 2022. GridBook: Natural Language Formulas for the Spreadsheet Grid. In *Proceedings of the 27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. Association for Computing Machinery, New York, NY, USA, 345–368. https://doi.org/10.1145/3490099.3511161

[36] Sruti Srinivasa Ragavan, Advait Sarkar, and Andrew D Gordon. 2021. Spreadsheet Comprehension: Guesswork, Giving Up and Going Back to the Author. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 181, 21 pages. https://doi.org/10.1145/3411764.3445634

[37] Yuta Takayama, Yuu Ichikawa, Buntarou Shizuki, Ikkaku Kawaguchi, and Shin Takahashi. 2021. A User-based Mid-air Hand Gesture Set for Spreadsheets. In *Asian CHI Symposium 2021 (Asian CHI '21)*. ACM, 122–128. https://doi.org/10.1145/3429360.3468193

[38] Theophanis Tsandilas, Anastasia Bezerianos, and Thibaut Jacob. 2015. SketchSliders: Sketching Widgets for Visual Exploration on Wall Displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3255–3264. https://doi.org/10.1145/2702123.2702129

[39] Theophanis Tsandilas, Catherine Letondal, and Wendy E. Mackay. 2009. Musink: composing music through augmented drawing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 819–828. https://doi.org/10.1145/1518701.1518827

[40] Theophanis Tsandilas and Wendy E. Mackay. 2010. Knotty gestures: subtle traces to support interactive use of paper. In *Proceedings of the International Conference on Advanced Visual Interfaces* (Roma, Italy) *(AVI '10)*. Association for Computing Machinery, New York, NY, USA, 147–154. https://doi.org/10.1145/1842993.1843020

[41] Huawei Tu, Xiangshi Ren, and Shumin Zhai. 2012. A comparative evaluation of finger and pen stroke gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1287–1296. https://doi.org/10.1145/2207676.2208584

[42] Barbara Tversky. 2002. What do sketches say about thinking. In *Sketch Understanding, papers from the 2002 AAAI Spring Symposium, March 25-27, 2002.* 148–151.

[43] Radu-Daniel Vatavu and Jacob O. Wobbrock. 2015. Formalizing Agreement Analysis for Elicitation Studies: New Measures, Significance Test, and Toolkit. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1325–1334. https://doi.org/10.1145/2702123.2702223

[44] Radu-Daniel Vatavu and Jacob O. Wobbrock. 2022. Clarifying Agreement Calculations and Analysis for End-User Elicitation Studies. *ACM Trans. Comput.-Hum. Interact.* 29, 1, Article 5 (2022). https://doi.org/10.1145/3476101

[45] Santiago Villarreal-Narvaez, Jean Vanderdonckt, Radu-Daniel Vatavu, and Jacob O. Wobbrock. 2020. A Systematic Review of Gesture Elicitation Studies: What Can We Learn from 216 Studies?. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*. Association for Computing Machinery, New York, NY, USA, 855–872. https://doi.org/10.1145/3357236.3395511

[46] Rand Wilcox. 2011. *Modern statistics for the social and behavioral sciences.* CRC Press, Boca Raton, FL.

[47] Jacob O. Wobbrock, Htet Htet Aung, Brandon Rothrock, and Brad A. Myers. 2005. Maximizing the guessability of symbolic input. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems* (Portland, OR, USA) *(CHI EA '05)*. Association for Computing Machinery, New York, NY, USA, 1869–1872. https://doi.org/10.1145/1056808.1057043

[48] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1083–1092. https://doi.org/10.1145/1518701.1518866

[49] Dustin Wüest, Norbert Seyff, and Martin Glinz. 2015. FlexiSketch Team: Collaborative Sketching and Notation Creation on the Fly. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE '15, Vol. 2)*. 685–688. https://doi.org/10.1109/ICSE.2015.223

[50] Haijun Xia, Michael Glueck, Michelle Annett, Michael Wang, and Daniel Wigdor. 2022. Iteratively Designing Gesture Vocabularies: A Survey and Analysis of Best Practices in the HCI Literature. *ACM Trans. Comput.-Hum. Interact.* 29, 4, Article 37 (May 2022), 54 pages. https://doi.org/10.1145/3503537

[51] Haijun Xia, Nathalie Henry Riche, Fanny Chevalier, Bruno De Araujo, and Daniel Wigdor. 2018. DataInk: Direct and Creative Data-Oriented Drawing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173797

[52] Haijun Xia, Ken Hinckley, Michel Pahud, Xiao Tu, and Bill Buxton. 2017. WritLarge: Ink Unleashed by Unified Scope, Action, & Zoom. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3227–3240. https://doi.org/10.1145/3025453.3025664

[53] Peng Xu, Timothy M. Hospedales, Qiyue Yin, Yi-Zhe Song, Tao Xiang, and Liang Wang. 2023. Deep Learning for Free-Hand Sketch: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2023), 285–312. https://doi.org/10.1109/TPAMI.2022.3148853

[54] Emanuel Zgraggen, Robert Zeleznik, and Philipp Eichmann. 2016. Tableur: Handwritten Spreadsheets. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 2362–2368. https://doi.org/10.1145/2851581.2892326

[55] Shumin Zhai, Per Ola Kristensson, Caroline Appert, Tue Haste Anderson, and Xiang Cao. 2012. Foundational Issues in Touch-Surface Stroke Gesture Design — An Integrative Review. *Foundations and Trends® in Human–Computer Interaction* 5, 2 (2012), 97–205. https://doi.org/10.1561/1100000012

# A Overview of all Sketch Categories

**Table 3: Breakdowns of categories for all tasks (excluding proposals missing command aspects or solving wrong tasks)**

| Task | Title | AR | A expr. | $P_1$ expr. | $P_2$ expr. | $P_3$ expr. | Proposals | Example |
|---|---|---|---|---|---|---|---|---|
| 1 | Change value | 0.22 | IMP | TXT | SEL | | 14 | Figure 20 |
| | | | IMP | TXT | LOC.P1 | | 11 | Figure 21 |
| | | | IMP | GUI | SEL | | 5 | |
| | | | IMP | CON.START | CON.END | | 2 | |
| | | | IMP | TXT | LOC.P1 | (2 steps) | 1 | |
| | | | SYM | TXT | LOC.A | | 1 | |
| | | | TXT | TXT | SEL | | 1 | |
| | | | CON | TXT | SEL | | 1 | |
| | | | SYM | TXT | TXT | | 1 | |
| | | | SYM | TXT | SEL | | 1 | |
| 2 | Delete value | 0.59 | IMP | SEL | | | 27 | Figure 22 |
| | | | GUI | SEL | | | 7 | |
| | | | TXT | TXT | | | 1 | |
| | | | CON.END | CON.START | | | 1 | |
| 3 | Delete values | 0.37 | IMP | SEL | (range) | | 21 | Figure 23 |
| | | | GUI | SEL | | | 8 | |
| | | | IMP | SEL | (cells) | | 4 | |
| | | | SYM | SEL | | | 1 | |
| | | | TXT | TXT | | | 1 | |
| | | | CON.END | SEL | | | 1 | |
| | | | TXT | SEL | | | 1 | |
| 4 | Add two values | 0.22 | TXT | TXT | TXT | LOC.A | 13 | Figure 24 |
| | | | SYM | SEL | SEL | LOC.A | 10 | Figure 25 |
| | | | GUI | SEL | SEL | SEL | 6 | |
| | | | TXT | SEL | SEL | LOC.A | 4 | |
| | | | SYM | LOC.A | LOC.A | SYM | 3 | |
| | | | IMP | SEL | SEL | CON.END | 1 | |
| 5 | Insert column | 0.22 | GUI | SEL | | | 12 | |
| | | | IMP | SEL | | | 9 | Figure 26 |
| | | | SYM | LOC.A | | | 9 | Figure 27 |
| | | | TXT | SEL | | | 3 | |
| | | | EXA | CON.END | | | 1 | |
| | | | EXA | LOC.A | | | 1 | |
| | | | TXT | TXT | | | 1 | |
| 6 | Remove column | 0.28 | IMP | SEL | (header) | | 16 | Figure 28 |
| | | | IMP | SEL | (range) | | 12 | Figure 29 |
| | | | GUI | SEL | | | 9 | |
| | | | CON.END | CON.START | | | 1 | |
| | | | TXT | SEL | | | 1 | |
| | | | TXT | TXT | | | 1 | |
| 7 | Move cells | 0.43 | CON | SEL | CON.END | | 21 | Figure 30 |
| | | | GUI | SEL | SEL | | 8 | |
| | | | CON+TXT | SEL | CON.END | | 2 | |
| | | | CON | CON.START | CON.END | | 2 | |
| | | | SYM | SEL | LOC.A | | 1 | |
| 8 | Add multiple values | 0.36 | SYM | SEL | LOC.A | | 20 | Figure 31 |
| | | | GUI | SEL | SEL | | 8 | |
| | | | TXT | TXT | LOC.A | | 5 | |
| | | | SYM | IMP | LOC.A | | 1 | |
| | | | SYM | SEL | SEL | | 1 | |
| | | | IMP | SEL | CON.END | | 1 | |
| 9 | Format values | 0.33 | IMP | SEL | GUI | | 16 | |
| | | | IMP | SEL | SYM | | 12 | Figure 32 |
| | | | TXT | SEL | SYM | | 5 | |
| | | | IMP | SEL | TXT | | 1 | |
| | | | IMP | SEL | EXA | | 1 | |
| 10 | Frame cell | 0.29 | GUI | SEL | | | 15 | |
| | | | IMP | SEL | | | 12 | Figure 33 |
| | | | SYM | SEL | | | 4 | |
| | | | TXT | SEL | | | 4 | |
| | | | SEL | SEL | | | 1 | |
| 11 | Transfer formatting | 0.22 | GUI | SEL | SEL | | 12 | |
| | | | CON+TXT | SEL | SEL | | 7 | Figure 34 |
| | | | TXT | SEL | SEL | | 5 | |
| | | | IMP | SEL | SEL | | 3 | |
| | | | SYM | SEL | SEL | | 1 | |
| | | | TXT | TXT | TXT | | 1 | |
| | | | TXT | TXT | SEL | | 1 | |
| | | | IMP | TXT | SEL | | 1 | |
| 12 | Create pie chart | 0.33 | GUI | SEL | | | 15 | |

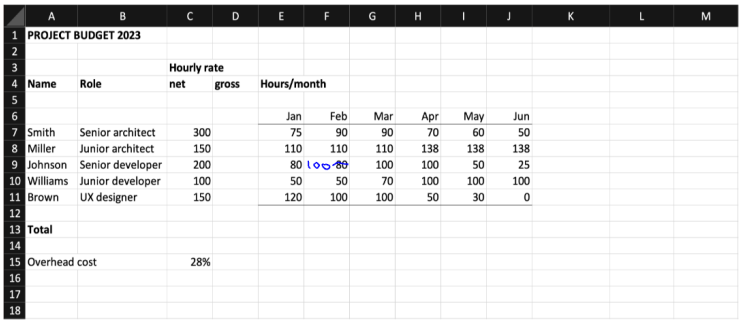
Continued on next page

Table 3 – continued from previous page

| Task | Title | AR | A expr. | $P_1$ expr. | $P_2$ expr. | $P_3$ expr. | Proposals | Example |
|---|---|---|---|---|---|---|---|---|
| | | | EXA | SEL | | | 14 | Figure 35 |
| | | | TXT | SEL | | | 5 | |
| | | | TXT | TXT | | | 2 | |
| 13 | Create bar chart | 0.31 | GUI | SEL | | | 15 | |
| | | | EXA | SEL | | | 14 | Figure 36 |
| | | | TXT | SEL | | | 7 | |
| | | | TXT | TXT | | | 2 | |
| 14 | Continue series | 0.39 | IMP | SEL | SEL | | 22 | Figure 37 |
| | | | GUI | SEL | SEL | | 5 | |
| | | | IMP | SEL | IMP | | 2 | |
| | | | TXT+EXA | SEL | SEL | | 2 | |
| | | | TXT | SEL | SEL | | 2 | |
| | | | SYM | SEL | SYM | | 1 | |
| | | | IMP | IMP | SEL | | 1 | |
| | | | SYM | SEL | SEL | | 1 | |
| 15 | Transpose values | 0.16 | GUI | SEL | SEL | | 12 | |
| | | | CON | SEL | SEL | | 6 | Figure 38 |
| | | | TXT | SEL | CON.END | | 5 | Figure 39 |
| | | | SYM | SEL | SEL | | 4 | Figure 40 |
| | | | IMP | SEL | SYM | | 2 | |
| | | | IMP | SEL | SEL | | 1 | |
| | | | SYM | SEL | CON.END | | 1 | |
| | | | TXT | TXT | LOC.A | | 1 | |
| | | | TXT | SEL | SEL | | 1 | |
| | | | EXA | SEL | LOC.A | | 1 | |
| 16 | Sort data | 0.23 | GUI | SEL | GUI | SEL | 16 | |
| | | | IMP | IMP | TXT | SEL | 5 | Figure 41 |
| | | | IMP | SEL | TXT | LOC.P2 | 5 | Figure 42 |
| | | | TXT | IMP | IMP | TXT | 2 | |
| | | | TXT | SEL | TXT | SEL | 2 | |
| | | | TXT | IMP | TXT | SEL | 2 | |
| | | | IMP | SEL | TXT | SEL | 1 | |
| | | | TXT | SEL | IMP | TXT | 1 | |
| | | | TXT | IMP | TXT | SEL+TXT | 1 | |
| | | | IMP | SEL | SYM | LOC.P2 | 1 | |
| 17 | Calculate multiple sums | 0.23 | SYM | SEL | SEL | (column) | 14 | Figure 43 |
| | | | GUI | SEL | SEL | | 9 | |
| | | | TXT | TXT | SEL | | 5 | |
| | | | SYM | SEL | LOC.A | | 4 | |
| | | | SYM | SEL | SEL | (table) | 3 | |
| | | | SYM | SEL | IMP | | 1 | |
| 18 | Construct formula | 0.12 | GUI | SEL | SEL | SEL | 9 | |
| | | | TXT | TXT | TXT | SEL | 7 | Figure 44 |
| | | | SYM | SEL | SEL | SEL | 5 | Figure 45 |
| | | | SYM | SEL | SEL | IMP | 2 | |
| | | | SYM | SEL | TXT | SEL | 2 | |
| | | | TXT | TXT | TXT | LOC.A | 2 | |
| | | | CON | SEL | SEL | SEL | 2 | |
| | | | TXT | SEL | SEL | SEL | 1 | |
| | | | SYM | CON.START | TXT | CON.END | 1 | |
| | | | SYM | SEL | SEL+TXD | TXT | 1 | |
| | | | SYM+CON | SEL | SEL | SEL | 1 | |
| | | | SYM | SEL | TXT | SEL | 1 | |
| | | | TXT | TXT | TXT | TXT | 1 | |
| 19 | Conditional formatting | 0.30 | GUI | SEL | GUI | GUI | 14 | |
| | | | IMP | SEL | TXT | TXT | 11 | Figure 46 |
| | | | TXT | SEL | TXT | TXT | 10 | |
| | | | TXT | TXT | TXT | TXT | 1 | |
| 20 | Multiply values | 0.16 | SYM | SEL | SEL | LOC.A | 8 | Figure 47 |
| | | | GUI | SEL | SEL | SEL | 8 | |
| | | | TXT | TXT | TXT | LOC.A | 7 | Figure 48 |
| | | | TXT | TXT | SEL | LOC.A | 6 | Figure 49 |
| | | | CON+TXT | SEL | SEL | CON.END | 4 | |
| | | | SYM+TXT | SEL | SEL | LOC.A | 2 | |
| | | | TXT | TXT | TXT | TXT | 1 | |

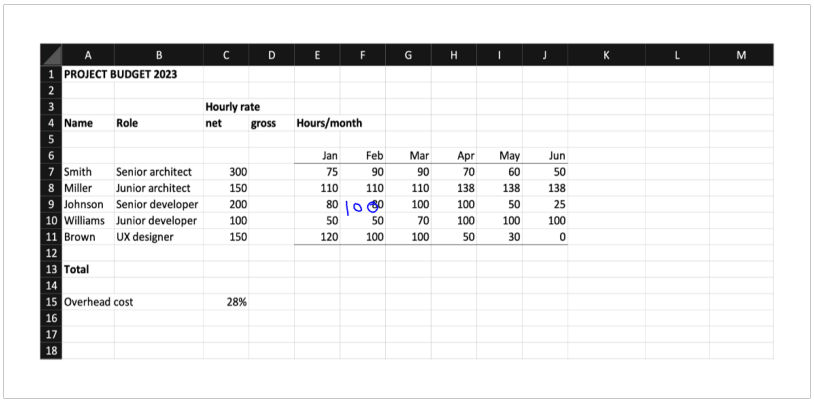
## B Example Sketches for all Tasks

For the sake of brevity, we only include example sketches for the largest categories not involving GUI elements in the paper. The complete repository of all participants' sketches is available from the authors upon request.



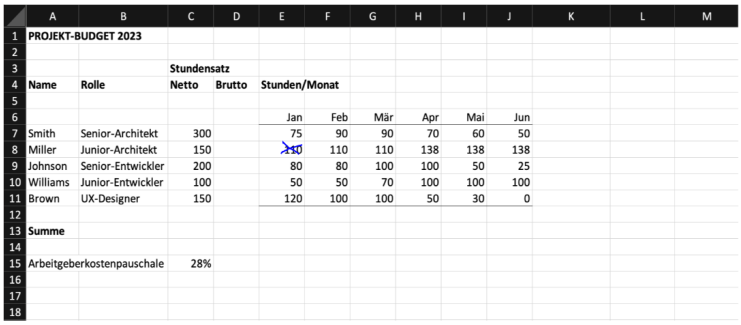
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 20: Task 1 (Change value) example for category (A:IMP, $P_1$:TXT, $P_2$:SEL)**



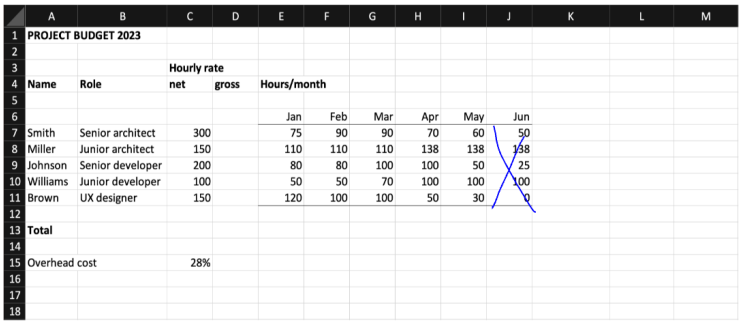
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 21: Task 1 (Change value) example for category (A:IMP, $P_1$:TXT, $P_2$:LOC.$P_1$)**



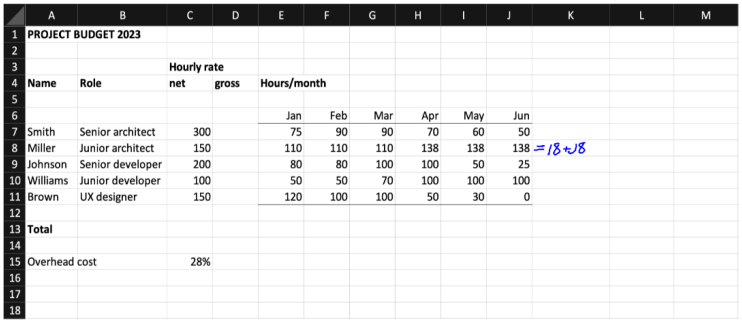
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 22: Task 2 (Delete value) example for category (A:IMP, $P_1$:SEL)**



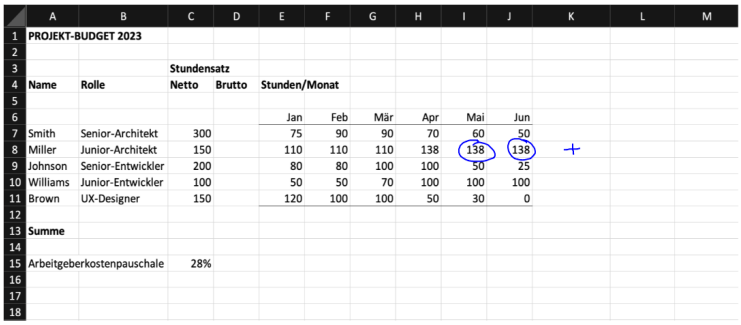
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 23: Task 3 (Delete values) example for category (A:IMP, $P_1$:SEL (range))**



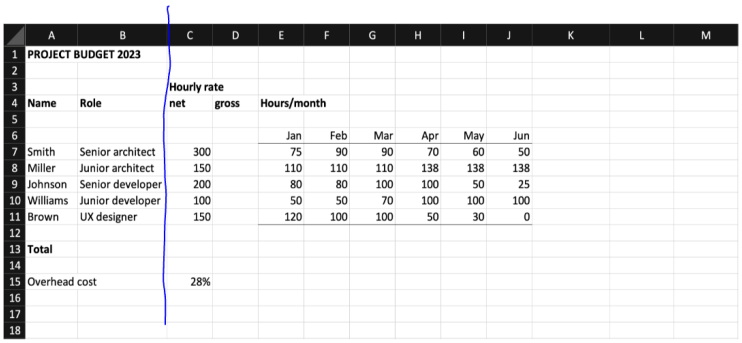
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | =J8+J8 | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 24: Task 4 (Add two values) example for category (A:TXT, $P_1$:TXT, $P_2$:TXT, $P_3$:LOC.A)**



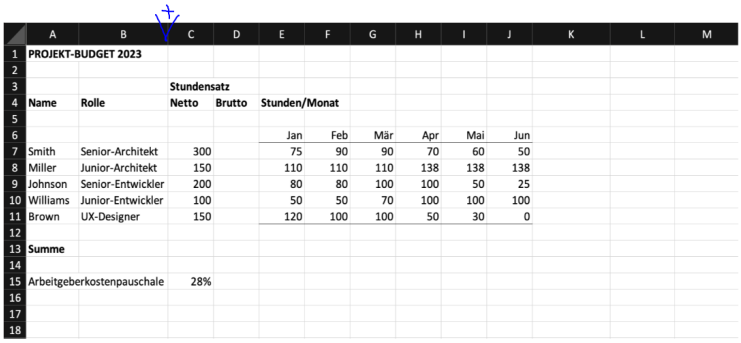
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | + | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 25: Task 4 (Add two values) example for category (A:SYM, $P_1$:SEL, $P_2$:SEL, $P_3$:LOC.A)**



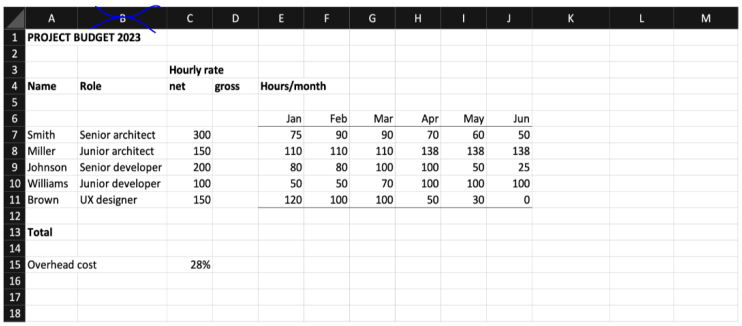
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 26: Task 5 (Insert column) example for category (A:IMP, $P_1$:SEL)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 27: Task 5 (Insert column) example for category (A:SYM, $P_1$:LOC.A)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 28: Task 6 (Remove column) example for category (A:IMP, $P_1$:SEL (header))**

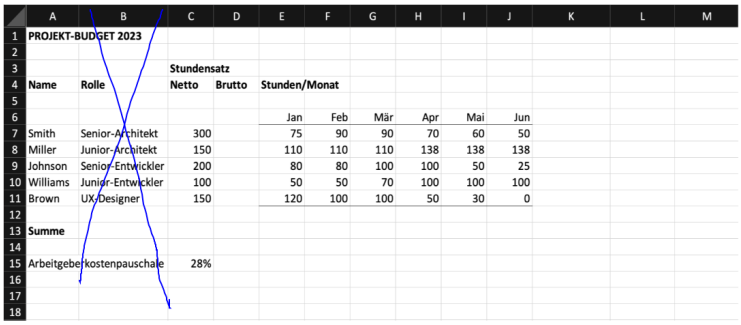



**Figure 29: Task 6 (Remove column) example for category (A:IMP, $P_1$:SEL (range))**

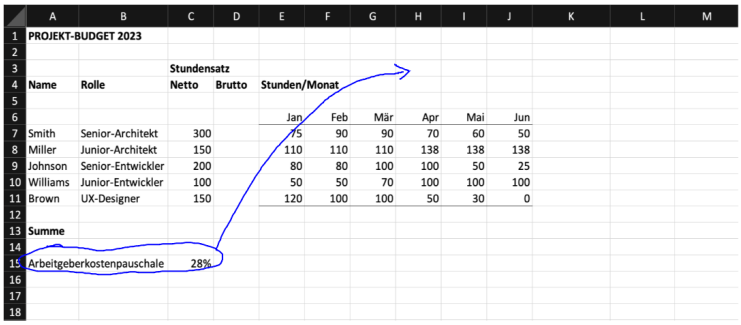



**Figure 30: Task 7 (Move cells) example for category (A:CON, $P_1$:SEL, $P_2$:CON.END)**

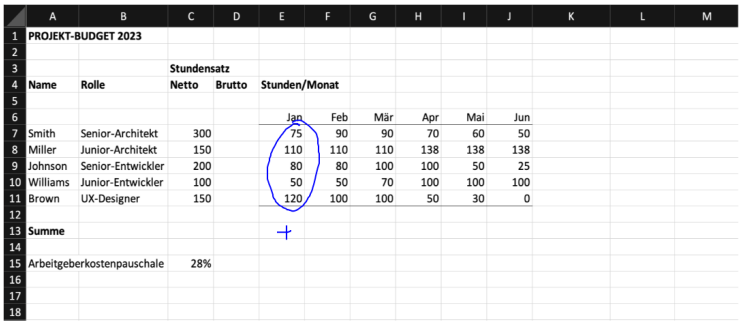
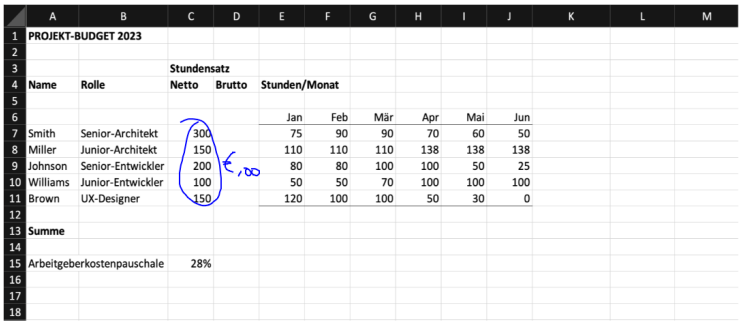
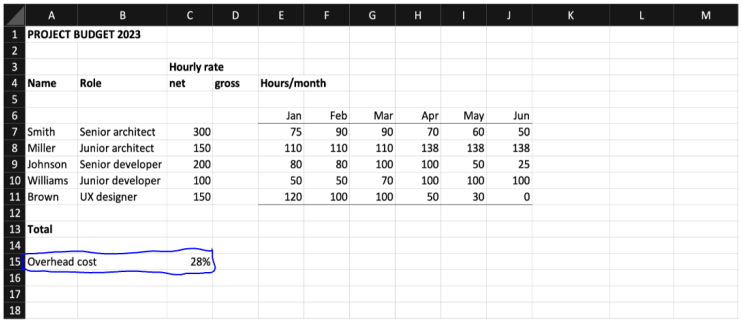



**Figure 31: Task 8 (Add multiple values) example for category (A:SYM, $P_1$:SEL, $P_2$:LOC.A)**



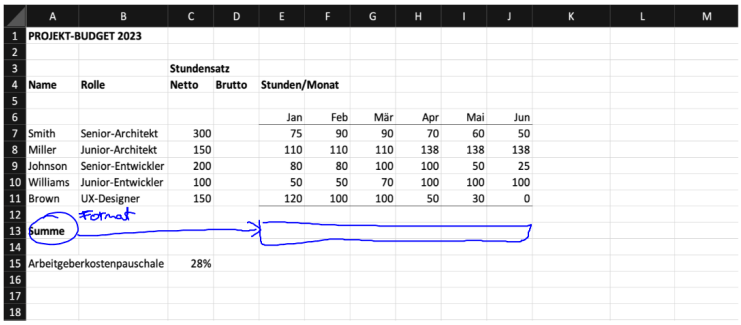
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 32: Task 9 (Format values) example for category (A:IMP, $P_1$:SEL, $P_2$:SYM)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 33: Task 10 (Frame cell) example for category (A:IMP, $P_1$:SEL)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 34: Task 11 (Transfer formatting) example for category (A:CON+TXT, $P_1$:SEL, $P_2$:SEL)**

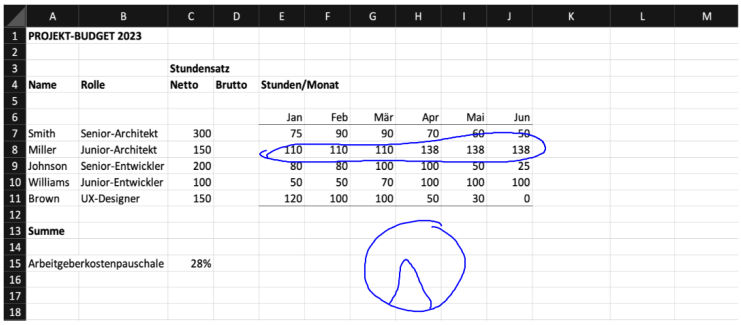



**Figure 35: Task 12 (Create pie chart) example for category (A:EXA, $P_1$:SEL)**

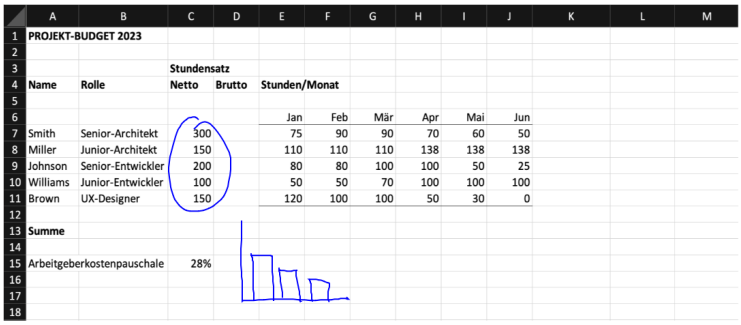



**Figure 36: Task 13 (Create bar chart) example for category (A:EXA, $P_1$:SEL)**

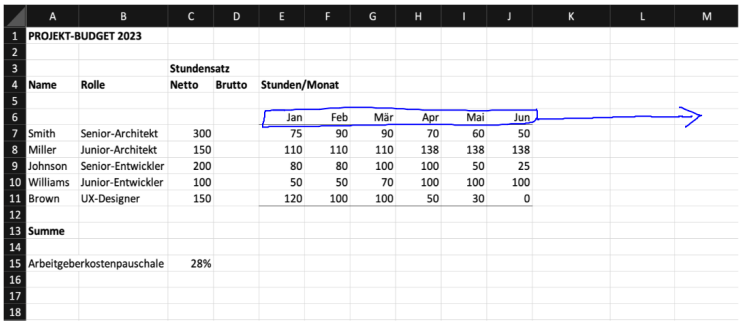
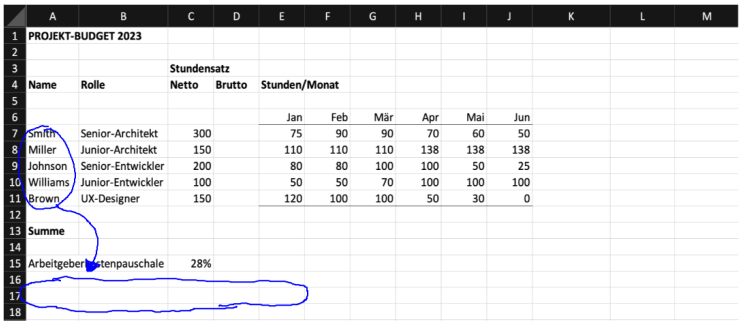
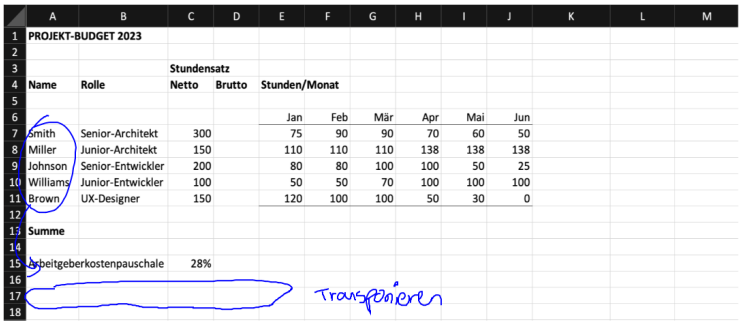



**Figure 37: Task 14 (Continue series) example for category (A:IMP, $P_1$:SEL, $P_2$:SEL)**



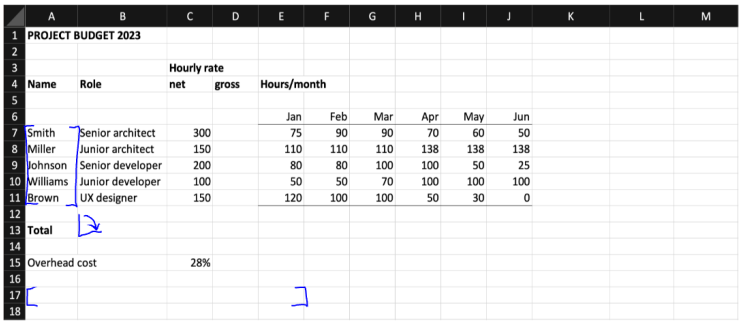
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 38: Task 15 (Transpose values) example for category (A:CON, $P_1$:SEL, $P_2$:SEL)**



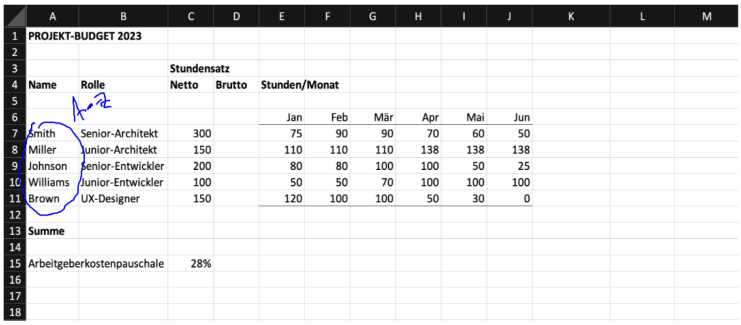
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 39: Task 15 (Transpose values) example for category (A:TXT, $P_1$:SEL, $P_2$:CON.END)**



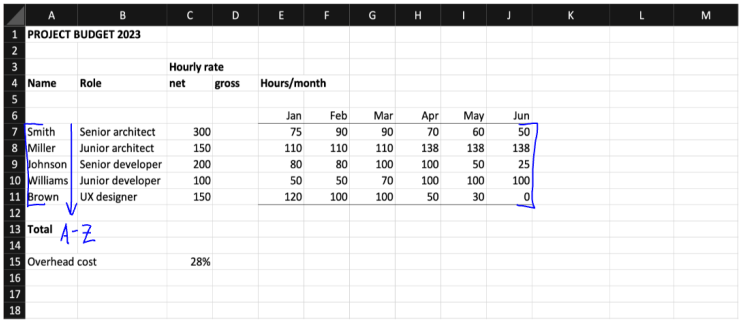
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun | | | |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 40: Task 15 (Transpose values) example for category (A:SYM, $P_1$:SEL, $P_2$:SEL)**



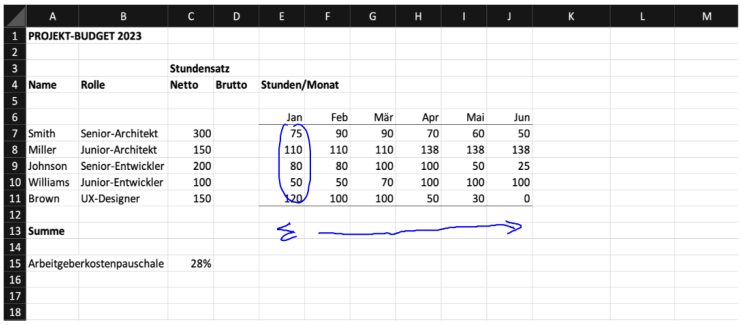

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | |
| 5 | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 |
| 12 | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | |
| 14 | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | | | | | | | | | | |
| 18 | | | | | | | | | | |

**Figure 41: Task 16 (Sort data) example for category (A:IMP, $P_1$:IMP, $P_2$:TXT, $P_3$:SEL)**




| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJECT BUDGET 2023** | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | | | **Hourly rate** | | | | | | | |
| 4 | **Name** | **Role** | **net** | **gross** | **Hours/month** | | | | | |
| 5 | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mar | Apr | May | Jun |
| 7 | Smith | Senior architect | 300 | | 75 | 90 | 90 | 70 | 60 | 50 |
| 8 | Miller | Junior architect | 150 | | 110 | 110 | 110 | 138 | 138 | 138 |
| 9 | Johnson | Senior developer | 200 | | 80 | 80 | 100 | 100 | 50 | 25 |
| 10 | Williams | Junior developer | 100 | | 50 | 50 | 70 | 100 | 100 | 100 |
| 11 | Brown | UX designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 |
| 12 | | | | | | | | | | |
| 13 | **Total** | | | | | | | | | |
| 14 | | | | | | | | | | |
| 15 | Overhead cost | | 28% | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | | | | | | | | | | |
| 18 | | | | | | | | | | |

**Figure 42: Task 16 (Sort data) example for category (A:IMP, $P_1$:SEL, $P_2$:TXT, $P_3$:LOC.$P_2$)**



| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | |
| 5 | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 |
| 12 | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | |
| 14 | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | | | | | | | | | | |
| 18 | | | | | | | | | | |

**Figure 43: Task 17 (Calculate multiple sums) example for category (A:SYM, $P_1$:SEL, $P_2$:SEL (column))**

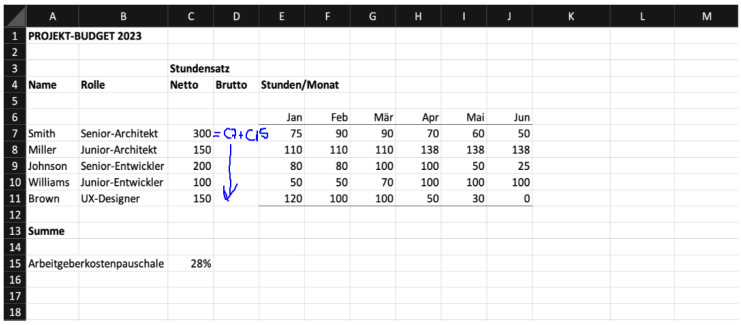



**Figure 44: Task 18 (Construct formula) example for category (A:TXT, $P_1$:TXT, $P_2$:TXT, $P_3$:SEL)**

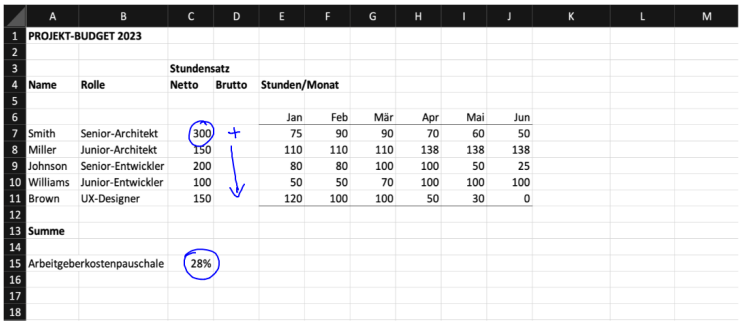



**Figure 45: Task 18 (Construct formula) example for category (A:SYM, $P_1$:SEL, $P_2$:SEL, $P_3$:SEL)**

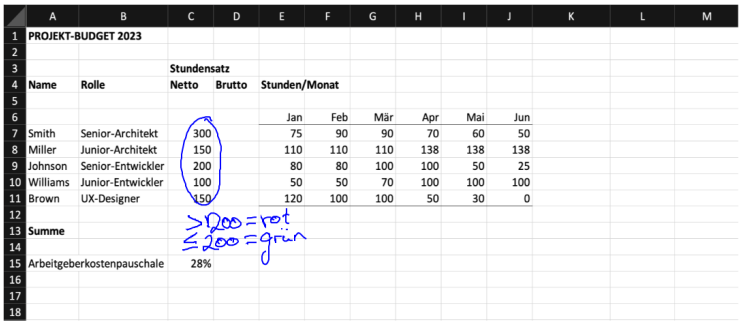
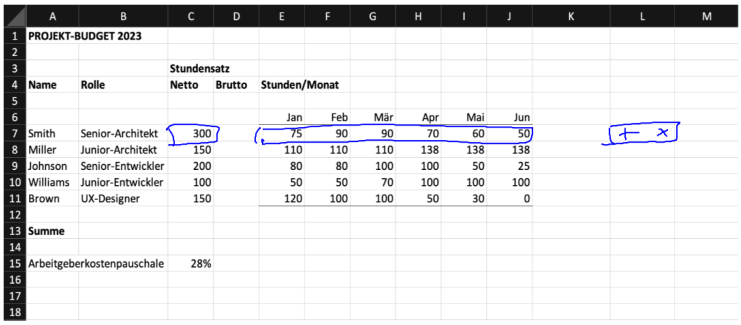
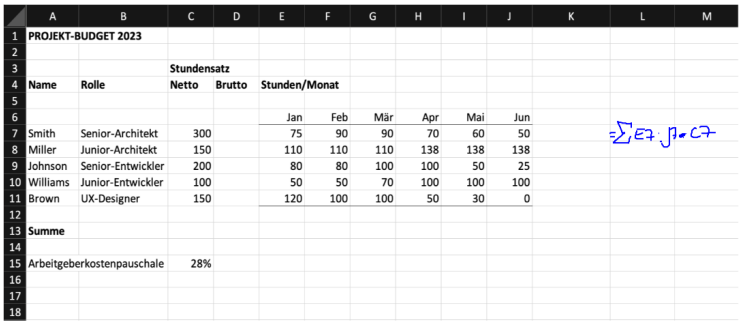



**Figure 46: Task 19 (Conditional formatting) example for category (A:IMP, $P_1$:SEL, $P_2$:TXT, $P_3$:TXT)**



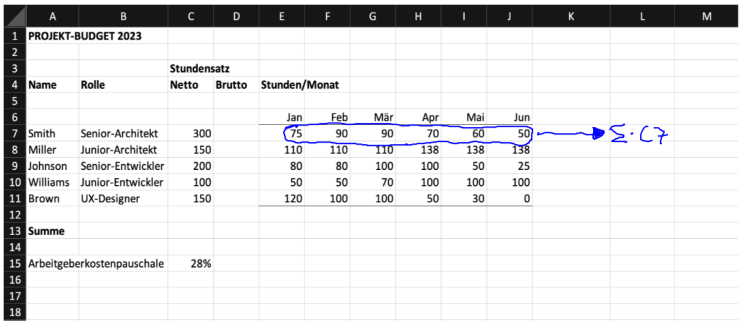
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 47: Task 20 (Multiply values) example for category (A:SYM, $P_1$:SEL, $P_2$:SEL, $P_3$:LOC.A)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 48: Task 20 (Multiply values) example for category (A:TXT, $P_1$:TXT, $P_2$:TXT, $P_3$:LOC.A)**



| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **PROJEKT-BUDGET 2023** | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | |
| 3 | | | **Stundensatz** | | | | | | | | | | |
| 4 | **Name** | **Rolle** | **Netto** | **Brutto** | **Stunden/Monat** | | | | | | | | |
| 5 | | | | | | | | | | | | | |
| 6 | | | | | Jan | Feb | Mär | Apr | Mai | Jun | | | |
| 7 | Smith | Senior-Architekt | 300 | | 75 | 90 | 90 | 70 | 60 | 50 | | | |
| 8 | Miller | Junior-Architekt | 150 | | 110 | 110 | 110 | 138 | 138 | 138 | | | |
| 9 | Johnson | Senior-Entwickler | 200 | | 80 | 80 | 100 | 100 | 50 | 25 | | | |
| 10 | Williams | Junior-Entwickler | 100 | | 50 | 50 | 70 | 100 | 100 | 100 | | | |
| 11 | Brown | UX-Designer | 150 | | 120 | 100 | 100 | 50 | 30 | 0 | | | |
| 12 | | | | | | | | | | | | | |
| 13 | **Summe** | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | |
| 15 | Arbeitgeberkostenpauschale | | 28% | | | | | | | | | | |
| 16 | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | |

**Figure 49: Task 20 (Multiply values) example for category (A:TXT, $P_1$:TXT, $P_2$:SEL, $P_3$:LOC.A)**