

Jan Schefer / Luk Schrodt / Mikka Kummer / Cedric Zollinger

Noser Young AG

8.12.2021

Taschenrechner in C

INHALTSVERZEICHNIS

[1 Informieren 2](#_Toc90450639)

[1.1 Projektauftrag 2](#_Toc90450640)

[1.2 Vorgaben 2](#_Toc90450641)

[1.2.1 Pflichtfeatures 2](#_Toc90450642)

[1.3 Zusätzliche Features 3](#_Toc90450643)

[1.3.1 2D Features 3](#_Toc90450644)

[1.3.2 3D Features 3](#_Toc90450645)

[1.3.3 Life-Science Features 3](#_Toc90450646)

[2 Planung 3](#_Toc90450647)

[2.1 Aufteilung 3](#_Toc90450648)

[2.2 Zeitmanagement 4](#_Toc90450649)

[2.3 Skizze Ablauf 4](#_Toc90450650)

[3 Entscheidung 5](#_Toc90450651)

[3.1 Ressourcen 5](#_Toc90450652)

[4 Realisierung 5](#_Toc90450653)

[4.1 Programmierung 5](#_Toc90450654)

[5 Kontrolle 6](#_Toc90450655)

[5.1 Planungskontrolle 6](#_Toc90450656)

[5.2 Funktionstest 6](#_Toc90450657)

[5.2.1 Testdrehbuch 6](#_Toc90450658)

[6 Auswertung 11](#_Toc90450659)

[6.1 Gruppenfazit 11](#_Toc90450660)

[6.2 Fazit Luk Schrodt 11](#_Toc90450661)

[6.3 Fazit Jan Schefer 12](#_Toc90450662)

[6.4 Fazit Cedric Zollinger 12](#_Toc90450663)

[6.5 Fazit Mikka Kummer 12](#_Toc90450664)

[7 Quellen 13](#_Toc90450665)

[7.1 Bildquellenverzeichnis 13](#_Toc90450666)

# Informieren

## Projektauftrag

In diesem Auftrag soll ein Taschenrechner mit der Programmiersprache C gebaut werden. Für diesen Auftrag gelten folgende Vorgaben:

## Vorgaben

* Das Projekt soll nach IPERKA umgesetzt werden.
* Das Projekt wird in zufällig gewählten 3er bis 4er Gruppen durchgeführt.
* Die Kollaboration soll vollkommen über GitHub stattfinden. Hierzu soll für das Projekt ein neues Repository erstellt und die Gruppenmitglieder und Ausbilder als Collaborators hinzugefügt werden.
* Die Features sollen sinngemäss in Funktionen aufgeteilt werden, welche auf verschiedene Dateien ausgelagert sind.
* Jedes Gruppenmitglied soll einen ungefähr gleichwertigen Bestandteil zur Arbeit beitragen.
* Pflichtfeatures sind als solche im Punkt 1.2.1markiert. Zusätzlich zu diesen soll zwischen einem der drei Wahlfeature-Blöcken gewählt werden. Für schnelle Gruppen können auch alle Features umgesetzt werden, anschliessend können auch eigene Wahlfeature-Blöcke realisiert werden.
* Für die Umsetzung der Features darf das Header File math.h **nicht** verwendet werden.
* Es wird eine Dokumentation erwartet (abgelegt auf Github). Diese enthält die einzelnen Schritte von IPERKA und deren Umsetzungen (siehe IPERKA\_Checkliste in der Einführungswoche).
* Der Code muss gemäss Definition of Done überarbeitet sein.
* Der Programmcode wird präsentiert (mind. 5 Minuten, max. 7 Minuten).

### Pflichtfeatures

Für alle Pflichtfeatures soll es die Möglichkeit geben, mit mehreren Zahlen zu arbeiten. (z.B. x+y+z) Die Operatoren werden hier (noch) nicht gemischt. (z.B. 3+ 4 \* 2 gehört nicht dazu) Es soll aber möglich sein, negative Zahlen zu multiplizieren.

Ein Bild, das Text, Whiteboard enthält.

Automatisch generierte Beschreibung•Summieren

•Subtrahieren

•Dividieren

•Multiplizieren

•Exponenten

•Wurzeln

•Fakultät

## Zusätzliche Features

### 2D Features

Ein Bild, das Text, Whiteboard enthält.

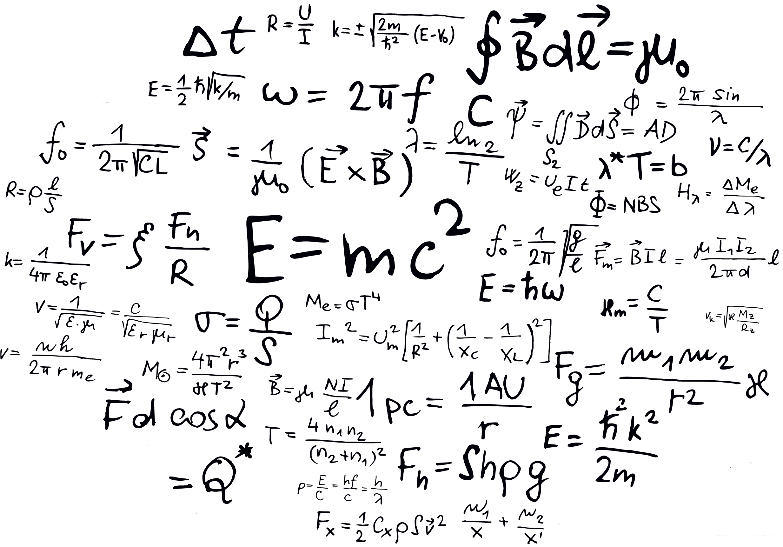
Automatisch generierte BeschreibungBerechnen der Fläche von folgenden Formen:

* Kreis
* Quadrat
* Rechteck
* Dreieck
* Trapez

### 3D Features

Berechnung des Volumens von folgenden Körpern:

* Kugel
* Würfel
* Quader
* Zylinder
* Kegel



### Life-Science Features

Weitere Berechnungen:

* BMI
* Promillerechner
* Kalorienbedarf-Rechner

# Planung

## Aufteilung

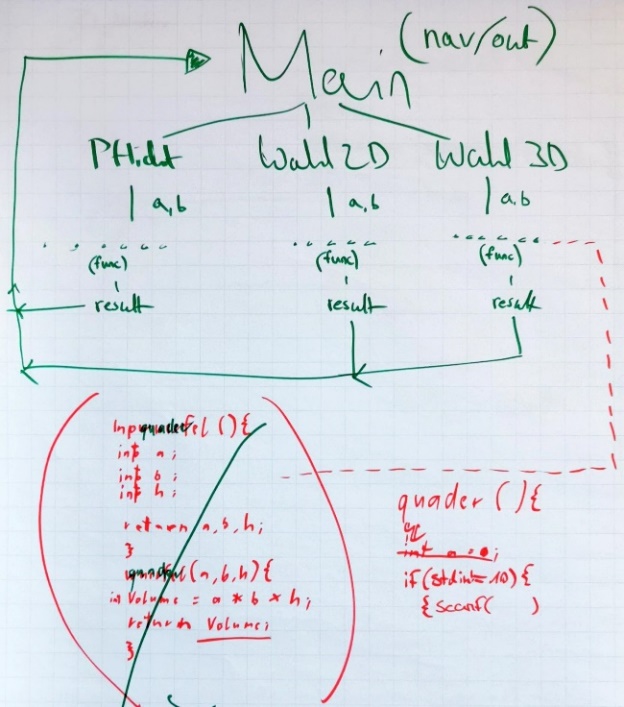
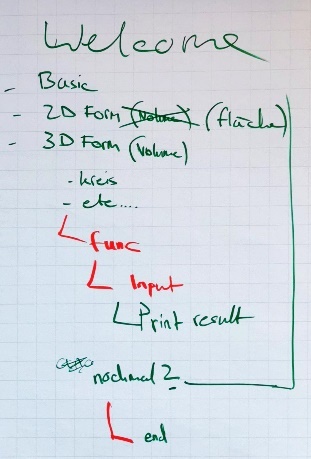
|  |  |
| --- | --- |
| Bereich | Verantwortlich |
| Führen und Aktualisieren der Dokumentation | J.Schefer / C.Zollinger |
| Bereitstellung der Hardware | Alle |
| Main Menü | M.Kummer |
| Pflicht Features | L.Schrodt |
| Wahl 2D Features | C.Zollinger |
| Wahl 3D Features | J.Schefer |
| Life Science Features | J.Schefer |
| Funktionstest und Kontrollen | L.Schrodt / J.Schefer |

## Zeitmanagement

|  |  |  |  |
| --- | --- | --- | --- |
| Tag | Ziele | Erreichte Arbeit | Kommentar |
| Mittwoch | Den Auftrag erfassen und verstehen Die Aufgaben aufteilen Fertigstellung des Main Erstellen des Repositories | Aufgabe wurde verstanden Main und Repository wurde fertiggestellt | Tagesziele konnten erfüllt werden |
| Donnerstag | Basic-Funktionen, 2D/3D, sowie Life-Science Funktionen fertigstellen | Alle Funktionen wurden beendet und getestet | Tagesziele konnten erfüllt werden Es blieb Zeit übrig für weitere Funktionen |
| Freitag | Dokumentation fertig stellen Zusätzliche Features Programmieren und einfügen | Dokumentation wurde fertig gestellt und zum Gegenlesen weitergeleitet. Zusätzliche Features wurden erstellt und im Code eingefügt | Tagesziele konnten erfüllt werden |

## Skizze Ablauf

Der Aufbau wird nach Ablauf der untenstehenden Skizze umgesetzt.



1. Main Menü  
2. Auswahl der Funktionsgruppen  
3. Auswahl der einzelnen Funktionen der jeweiligen Gruppe  
4. Ausführen des Codes und Berechnen der Formel  
5. Das Resultat der Formel Ausgeben und mittels <weiter> zurück ins Main Menü gelangen.

1

2

3

5

4

# Entscheidung

## Ressourcen

In der Ressourcentabelle werden alle benutzen Software, Hardware, sowie Infrastrukturen aufgelistet. In der Spalte Verwendung wird unsere Verwendung des jeweiligen Produktes kurz erklärt.

|  |  |  |
| --- | --- | --- |
| Kategorie | Produkt | Verwendung |
| Software | C-Lion | Die zur Erstellung des Codes |
| Software | GitHub | Teilen und Zusammenfügen von Dokumenten und Codes |
| Software | Word | Erstellung des Auftrages, sowie des Projektauftrages |
| Software | Microsoft-Teams | Kommunikation und Übermittlung von Bildern und Daten |
| Infrastruktur | WLAN | Allgemeine Nutzung |
| Infrastruktur | Laptop (privat) | Erstellen von Dokumenten und Programmieren des C-Codes |
| Infrastruktur | Arbeitsplatz | Allgemeine Nutzung |
| Hardware | Flipchart + Marker | Planung und Aufteilung |

# Realisierung

## Programmierung

Bei der Programmierung des ganzen Projekts begannen wir zuerst alle an unterschiedlichen Dateien, die wir alle regelmässig auf GitHub hochgeladen haben. Dabei machten wir schnell grosse Fortschritte und konnten noch am ersten Tag die Dateien zusammenführen. Dabei gab es leider mehrere Komplikationen. Diese konnten jedoch auch gelöst werden. Nach dem Zusammenführen arbeiteten teils auch mehr als nur eine Person an einem Code. Am zweiten Tag begannen wir dann mit dem genaueren Testen, wobei auch noch mehrere fatale Fehler auftraten, welche wir glücklicherweise bewältigen konnten. Da wir aber zu diesem Zeitpunkt oft doppelte Codes hatten und diese auch nicht defensiv programmiert waren, mussten wir dies noch nachholen. Das defensive Programmieren korrigierte jede Person an ihrem eigenen Code, das Auslagern in Funktionen konnte nur eine Person übernehmen. Grund dafür war, dass es hauptsächlich zwei Codezeilen waren, die sich immer wieder wiederholten. Anschliessend waren wir mit der «Definition of Done» fertig.

# Kontrolle

## Planungskontrolle

Im Abschnitt Planungskontrolle werden Tests analysiert, welche in Verlauf des Projektauftrages benutzt wurden. Die Tests variieren je nach Projekt. Es benötigt zudem verschiedene Tests, um ein flächendeckendes Resultat zu erreichen.

## Funktionstest

Funktionstest umschreibt eine Auswahl von verschiedenen Tests, welche bei einem Projekt zum Einsatz gekommen sind.

### Testdrehbuch

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Testfall-Nr. | | | 1 | | | |
| Testfall-Bezeichnung | | | Basic-Funktionen | | | |
| Anforderungs-Nr. | | | 1 | | | |
| Testumgebung | | | Noser Young | | | |
| Zu testende Funktionalität | | | Die Funktionen der Basic-Funktion auf Minus, Plus und Symbole testen. Weiter und Beenden testen. | | | |
| Datum der Testdurchführung | | | 09.12.2021 | | | |
| Tester | | | Jan Schefer, Luk Schrodt, Mikka Kummer, Cedric Zollinger | | | |
| Testschritte: | | | | | | |
| Nr. | Aktion | Erwartetes Ergebnis | | Effektives Ergebnis | Erfüllt | Kommentar |
| 1 | Summieren: -3 +3 | Ausgabe von 0 | | Anfrage für eine zweite Zahl fehlt | Nein | Code kann keine Minuszahlen entgegennehmen |
| 2 | Summieren: 2 + 3 | Ausgabe von 5 | | Es wurde kein Resultat geprintet | Jein | Die Print-Funktion fehlte, jedoch konnte im Debugger 5 nachgewiesen werden |
| 3 | Summieren: zd + 7 | Ausgabe von Fehlermeldung | | Abfrage, ob man <weiter> machen möchte oder <beenden> | Nein | Error Handling für Eingabe von Symbolen |
| 4 | Multiplizieren: 4 und 6 | Ausgabe von 24 | | Ausgabe von 24 | Ja | Erwartetes Resultat erhalten |
| 5 | Exponentiell: 4^5 | Ausgabe von 1024 | | Ausgabe von 1024 | Ja | Erwartetes Resultat erhalten |
| 6 | Subtrahieren: 50--3 | Ausgabe von 53 | | Ausgabe von 53 | Ja | Erwartetes Resultat erhalten |
| 7 | Summieren: 12+ -6 | Ausgabe von 6 | | Ausgabe von 6 | Ja | Erwartetes Resultat erhalten |
| 8 | Summieren: 24 + b | Ausgabe von Fehlermeldung | | Ausgabe von Fehlermeldung | Ja | Erwartetes Resultat erhalten |
| 9 | Wurzel: √3 | Ausgabe von 1.732 | | Ausgabe von 1.732 | Ja | Erwartetes Resultat erhalten |
| 10 | Wurzel: √-5 | Ausgabe von Fehlermeldung | | Ausgabe von Endlos-Schleife | Nein | Minuszahlen sollten nicht erlaubt werden |
| 11 | Fakultät: !12 | Ausgabe von 479001600 | | Ausgabe von 479001600 | Ja | Erwartetes Resultat erhalten |
| 12 | Wurzel: √-5 | Ausgabe von Fehlermeldung | | Ausgabe von Fehlermeldung | Ja | Erwartetes Resultat erhalten |
| 13 | Starten des Programms mit dem Input 4 | Das Programm rechnet die eigegebene Zahl hoch 0.5 und erhält 2 | | Das Programm rechnet das Ergebnis hoch 1 | Nein | Das Programm kann nur ganze Potenzen ausrechen |
| 14 | Starten des Programms mit neuem Code, der aufs Testen ausgelegt ist | Das Programm rechnet die eine Zahl immer plus 0.000001 bis diese Zahl, im Quadrat, die eingegebene Zahl überschreitet | | Das Programm gibt die Zahl zu ungenau aus | Nein | Die Ausgabe Variable muss vergrössert werden |
| 15 | Starten des Programms mit neuem Code, der aufs Testen ausgelegt ist | Das Programm rechnet die eine Zahl immer plus 0.000001 bis diese Zahl, im Quadrat, die eingegebene Zahl überschreitet | | Das Programm rechnet die eine Zahl immer plus 0.000001 bis diese Zahl, im Quadrat, die eingegebene Zahl überschreitet | Jein | Das Programm kann sehr genaue Angaben machen, aber bei Zahlen über einer Million dauert das Ausrechnen mindestens zehn Minuten |
| 16 | Starten des Programms mit Code, der eine Rechnung ausführt die eine Wurzel ausrechnen kann | Das Programm rechnet die Wurzel aus und gibt sie auf sechs Stellen genau aus | | Es gibt einen Error auf Grund einer Fehlenden Variable | Nein | Die Variable kann einfach hinzugefügt werden und sollte keine weiteren Probleme verursachen |
| 17 | Starten des Programms mit Code, der eine Rechnung ausführt, die eine Wurzel ausrechnen kann | Das Programm rechnet die Wurzel aus und gibt sie auf sechs Stellen genau aus | | Das Programm rechnet die Wurzel direkt aus und gibt sie auf sechs Stellen genau aus | Ja | Finale Lösung, da diese Methode schnell und genau ist |
| 18 | Fakultät: Starten des Programms | Abfrage zweier Zahlen und ausrechen des Ergebnisses | | Error, auf Grund eines Syntax Fehlers. | Nein | Es war ein nicht vorhandener Operator angegeben |
| 19 | Fakultät: Starten des Programms mit ersetztem Operator | Abfrage zweier Zahlen und ausrechen des Ergebnisses | | Abfrage zweier Zahlen und ausrechen des Ergebnisses | Jein | Es ist nur mit der math.h library möglich diesen Operator zu verwenden |
| 20 | Starten des Programms mit eigener Funktion für die Potenzierung | Abfrage zweier Zahlen und ausrechen des Ergebnisses | | Abfrage zweier Zahlen und ausrechen des Ergebnisses | Ja | Erwartetes Resultat erhalten |
| 21 | Starten des Programms mit Exponent Null | Ausgabe von 1 | | Ausgabe von 1 | Ja | Erwartetes Resultat erhalten |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Testfall-Nr. | | | 2 | | | |
| Testfall-Bezeichnung | | | 2D Funktionen | | | |
| Anforderungs-Nr. | | | 1 | | | |
| Testumgebung | | | Noser Young | | | |
| Zu testende Funktionalität | | | Die Funktionen zur Flächenberechnung verschiedener Formen | | | |
| Datum der Testdurchführung | | | 09.12.2021 | | | |
| Tester | | | Cedric Zollinger | | | |
| Testschritte: | | | | | | |
| Nr. | Aktion | Erwartetes Ergebnis | | Effektives Ergebnis | Erfüllt | Kommentar |
| 1 | Kreis: Minuszahl eingegeben | Berechnete Fläche | | Fläche des Kreises wurde ausgegeben | Ja | Erwartetes Resultat erhalten |
| 2 | Quadrat: Minuszahl  eingegeben | Berechnete Fläche im Minusbereich | | Fläche im Minusbereich | Ja | Erwartetes Resultat erhalten |
| 3 | Durchtesten aller Funktionen mit Eingabe von Buchstaben bei Flächenberechnungen | Falscheingabe Nachricht und wiederholende Abfrage | | Falscheingabe Nachricht und wiederholende Abfrage | Ja | Erwartetes Resultat erhalten |
| 4 | Durchtesten aller Funktionen mit Falscheingaben | Keine Fehler | | Keine Fehler im ersten Durchlauf | Ja | Aufgefallen ist jedoch, dass beim zweiten Durchlauf jeweils die Eingabe ausgedruckt wird direkt nach der Eingabe. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Testfall-Nr. | | | 3 | | | |
| Testfall-Bezeichnung | | | 3D Features | | | |
| Anforderungs-Nr. | | | 3 | | | |
| Testumgebung | | | Noser Young | | | |
| Zu testende Funktionalität | | | Testen der Funktionen der 3D | | | |
| Datum der Testdurchführung | | | 09.12.2021 | | | |
| Tester | | | Jan Schefer | | | |
| Testschritte: | | | | | | |
| Nr. | Aktion | Erwartetes Ergebnis | | Effektives Ergebnis | Erfüllt | Kommentar |
| 1 | Kugel: -3  3 | -3 = Fehlermeldung 3 = 14 | | -3 = Fehlermeldung 3 = 14 | Ja | Erwartetes Resultat erhalten |
| 2 | Würfel: -3 3 | -3 = Fehlermeldung 3 = 27 | | -3 = Fehlermeldung 3 = 27 | Ja | Erwartetes Resultat erhalten |
| 3 | Quader: -3 & 3 & 3 3 & -3 & 3 3 & 3 & -3 3 & 4 & 5 | Jeder Test mit einer  -3 = Fehlermeldung 3 & 4 & 5 = 60 | | Jeder Test mit einer  -3 = Fehlermeldung 3 & 4 & 5= 60 | Ja | Erwartetes Resultat erhalten |
| 4 | Zylinder: -3 & 3 3 & -3 3 & 3 | Jeder Test mit einer  -3 = Fehlermeldung 3 & 3 = 21 | | Jeder Test mit einer  -3 = Fehlermeldung 3 & 3 = 21 | Ja | Erwartetes Resultat erhalten |
| 5 | Kegel: -3 & 3 3 & -3 3 & 3 | Jeder Test mit einer  -3 = Fehlermeldung 3 & 3 = 5 | | Jeder Test mit einer  -3 = Fehlermeldung 3 & 3 = 5 | Ja | Erwartetes Resultat erhalten |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Testfall-Nr. | | | 4 | | | |
| Testfall-Bezeichnung | | | Life Science | | | |
| Anforderungs-Nr. | | | 1 | | | |
| Testumgebung | | | Noser Young | | | |
| Zu testende Funktionalität | | | Testen der Funktionen des BMI, Promillerechner, sowie dem Kalorienrechner | | | |
| Datum der Testdurchführung | | | 09.12.2021 | | | |
| Tester | | | Jan Schefer, Mikka Kummer | | | |
| Testschritte: | | | | | | |
| Nr. | Aktion | Erwartetes Ergebnis | | Effektives Ergebnis | Erfüllt | Kommentar |
| 1 | BMI: Grösse = -1.5 Gewicht = 50 | Jegliche Eingaben von Minuszahlen = Fehlermeldung | | Fehlermeldung | Ja | Erwartetes Resultat erhalten |
| 2 | BMI: Grösse = 1.80 Gewicht = 60 | Ausgabe von 19 | | Ausgabe von 19 | Ja | Erwartetes Resultat erhalten |
| 3 | Promillerechner: Geschlecht = m Alkohol = 50 Gewicht = 50 | Ausgabe von 1 | | Ausgabe von 1 | Ja | Erwartetes Resultat erhalten |
| 4 | Promillerechner: Geschlecht = s Alkohol = 50 Gewicht = 50 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 5 | Promillerechner: Geschlecht = f Alkohol = -50 Gewicht = 50 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 6 | Promillerechner: Geschlecht = f Alkohol = 50 Gewicht = -50 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 7 | Kalorienbedarf: Geschlecht = f Alter = 50 Gewicht = 60 Grösse = 180 | Ausgabe von 1320 Kalorien | | Ausgabe von 1320 Kalorien | Ja | Erwartetes Resultat erhalten |
| 8 | Kalorienbedarf: Geschlecht = s Alter = 50 Gewicht = 60 Grösse = 180 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 9 | Kalorienbedarf: Geschlecht = M Alter = -50 Gewicht = 60 Grösse = 180 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 10 | Kalorienbedarf: Geschlecht = M Alter = 50 Gewicht = -60 Grösse = 180 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 11 | Kalorienbedarf: Geschlecht = F Alter = 50 Gewicht = 60 Grösse = -180 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |
| 12 | Kalorienbedarf: Geschlecht = s Alter = 50 Gewicht = g Grösse = 180 | Fehlermeldung und Wiederholung der Eingabe | | Fehlermeldung und Wiederholung der Eingabe | Ja | Erwartetes Resultat erhalten |

# Auswertung

## Gruppenfazit

Wir waren insgesamt alle sehr zufrieden mit diesem Projekt. Wir hatte eine gute Planung und Gruppenaufteilung. Wir haben es geschafft, alle Teammitglieder in das Projekt miteinzubinden und den Auftrag mit fast allen Zusatzaufgaben abzuschliessen. Es gab leider auch ein paar Downs bei unserer Arbeit. Wir hatten unglücklicherweise zwei Mergeconflicts, aber selbst die konnten wir mit ein bisschen Engagement beheben. Rückblickend hätte man vielleicht an machen stellen die Kommunikation verbessern können, um diese nächsten Male von Anfang an zu verhindern. Dennoch denken wir, dass wir ein vernünftiges Endprodukt erstellt haben, das die Anforderungen erfüllt.

## Fazit Luk Schrodt

Meiner Meinung nach lief das Projekt äusserst gut. Wir kamen schnell voran und hatten eine hervorragende Planung. Ich fand auch meinen Teil des Auftrags sehr interessant und war interessiert daran, diesen Code zu vervollständigen. Ich habe dabei allerdings gelernt, dass ich öfter Tests durchführen sollte, damit ich Fehler früher finden und beheben kann. Ich bin trotzdem sehr glücklich über das Endresultat.

## Fazit Jan Schefer

Das Projekt Taschenrechner lief im Grossen und Ganzen wie geplant. Wir hatten wenige Probleme und können ein solides Programm vorweisen. Die Arbeit in der Gruppe hat Spass gemacht und wir konnten an unserer Kommunikation arbeiten. Ich bin froh, dass wir am Anfang des Projektes eine Formelsammlung erstellt haben, da sonst jeden Tag die Frage aufgekommen wäre, wie was ausgerechnet wird. Die Aufteilung in der Gruppe funktionierte auch wie geplant, obwohl es zwischenzeitlich zu Mergeconflicten kam.

## Fazit Cedric Zollinger

An dem Projekt zu arbeiten hat Spass gemacht. Wir konnten uns schnell als eine Gruppe finden und direkt beginnen, wie wir das Projekt angehen wollten. Wir haben uns schnell informiert und jeder hat seinen Teil zur Planung beigegeben. Die Aufteilung der Aufgaben haben wir auch glücklicherweise ohne grosse Komplikationen gemacht und so konnten wir Effizient an dem Auftrag arbeiten. Bei Schwierigkeiten halfen wir uns gegenseitig und wir kamen so schnell voran. Für uns war jedoch neu, dass wir über Git unseren Fortschritt übertragen haben. Anfangs lief das ohne Probleme da jeder seine Datei hatte, an der er gearbeitet hat, aber mit der Zeit fing es an, dass zwei Personen gleichzeitig eine Datei bearbeiten. Unglücklicherweise kam es dann noch zwischenzeitlich zu einem Mergeconflict, bei dem es mir die CMake Dateien mit Comitted hat. Dadurch Funktionierte der Compiler nicht mehr bei den anderen, da in der Datei Absolute Pfade meines Laptops enthalten waren. Aber durch ein bisschen Unterstützung konnten wir den Konflikt lösen und frisch und munter am nächsten Morgen weiter daran arbeiten. Bei der Überprüfung der Dokumente und dem Code konnten wir uns auch gut aufteilen. Den Code haben wir zusammen mit der «Definition of Done» überprüft und bearbeitet. Unsere Dokumentation konnten wir so auch nochmals auf Fehler überprüfen und zusammen durchgehen. Alles in Allem hatten wir als Team eine gute Strukturierung. Jeder wusste was er zu liefern hatte und wir konnten uns gegenseitig bei Problemen immer helfen und uns auch mal gegenseitig Konstruktive Kritik geben.

## Fazit Mikka Kummer

Die Arbeit an diesem Projekt war sehr spannend und hat tatsächlich sehr viel Spass gemacht. Durch dieses Projekt habe ich gemerkt, dass C Programmieren doch nicht so schlimm ist, wie bisher gedacht.

Wir haben gut angefangen mit unserer Planung und Formelsammlung und konnten von dort aus gut weiterarbeiten. Die Teamarbeit lief relativ gut bis auf den einten Mergeconflict. Unsere Arbeit war gut aufgeteilt und wir sind schnell fertig geworden.

Unser Endprodukt ist sicherlich nicht das Beste in diesem Raum, aber trotzdem kann ich behaupten, dass es sehr solide und funktional ist.

# Quellen

## Bildquellenverzeichnis

Abbildung Titelbild

Zuletzt aufgerufen: 08.12.2021

<https://www.pngegg.com/de/png-byoxw>

Abbildung Formeln

Zuletzt aufgerufen: 08.12.2021

<https://www.pngfind.com/mpng/iTiRxJb_math-formulas-vector-math-formula-hd-png-download/>