Wirtschaft
Hauptcampus

H O C H
S C H U L E
T R I E R

# SOMMERSEMESTER 2023

| | |
|---|---|
| **Studiengang/Abschluss:** | Wirtschaftsinformatik / B.Sc. |
| **Prüfungsfach:** | Grundlagen der Programmierung |
| ***Prüfungsnummer:*** | ***5701*** |
| **Prüfer:** | Prof. Dr. Andreas Biesdorf |
| **Anzahl der Arbeitsblätter:** | **8** Seiten (ohne Deckblätter) |
| **Bearbeitungszeit:** | **90** Minuten |

*Mit Bleistift oder in roter Farbe geschriebene Ausführungen werden nicht gewertet!*

## *Zugelassene Hilfsmittel:*

Eine einseitig handbeschriebene DIN A4-Seite Python-Befehle oder -Syntax (z.B. die Syntax einer for- oder while-Schleife), jedoch keine Theorie oder Strukturen konkreter Implementierungen kompletter Aufgaben.

**Matrikelnummer:** ☐☐☐.☐☐☐

**Datum der Klausur:** …… **Platznummer:** ……

**Punkte:** _____ von **90** möglichen

**Note:** _____

**Handzeichen des Prüfers:** _____

Wirtschaft
Hauptcampus

H O C H
S C H U L E
T R I E R

## Hinweise zu Klausuren im FB Wirtschaft

- Mit Aufnahme der Prüfung bringen Sie zum Ausdruck, dass Sie sich für prüfungstauglich halten. Ein Rücktritt von der Klausur während der Bearbeitungszeit ist damit nur noch in ganz besonderen Ausnahmefällen möglich.
- Sollten Sie zu Beginn der Bearbeitungszeit prüfungstauglich sein, dann aber während der Bearbeitungszeit prüfungsuntauglich werden, müssen Sie die Klausur abbrechen, Ihre Prüfungsuntauglichkeit den Aufsichtführenden anzeigen und schnellstmöglich, in jedem Fall aber fristgerecht für ein ärztliches Attest sorgen.
- Mit der Abgabe Ihrer Klausur bringen Sie zum Ausdruck, dass Sie sich während der gesamten Bearbeitungszeit für prüfungstauglich gehalten haben. Ein Rücktritt ist dann nicht mehr möglich.
- Jacken und Taschen sind in ausreichender Entfernung vom Platz zu deponieren.
- Die Bearbeitungszeit beginnt erst, wenn alle Klausuren ausgeteilt sind.
- Es sind nur die zugelassenen Hilfsmittel zu verwenden.
- Das Verlassen des Raumes für Toilettengänge während der Klausur ist nur nach Meldung und Bestätigung der Aufsicht gestattet. Die vorübergehende Abwesenheit wird im Protokoll vermerkt.
- Es sind keine anderen als die ausgegebenen Bearbeitungsblätter zu benutzen.
- Auf dem Tisch sind nur Schreibutensilien erlaubt, keine Mäppchen.
- Jeder Teilnehmer muss sich mit einem Studierendenausweis oder Lichtbildausweis ausweisen.
- Geräusche und Störungen jeder Art sind zu vermeiden.
- Die Klausurensätze sind grundsätzlich nicht zu öffnen. Wer den Klausurensatz öffnet, trägt dafür Sorge, dass die Blätter (innerhalb der Bearbeitungszeit) sortiert und neu geheftet werden. Das Risiko des Verlustes von einzelnen Blättern trägt der Studierende!
- Es ist nicht erlaubt, Aufgabenstellungen auf gesondertes Papier abzuschreiben.
- Mobiltelefone, Smartphones und Smartwatches sind nicht erlaubt!
- Alle Studierenden bleiben auf ihren Plätzen und verhalten sich ruhig, bis die Klausuren vollständig eingesammelt sind. Folgen Sie den Anweisungen der Aufsichtsführenden.
- Bei Täuschung oder versuchter Täuschung wird mit „nicht ausreichend“ bewertet.

Wirtschaft
Hauptcampus

H O C H
S C H U L E
T R I E R

Die Klausur besteht aus 8 Aufgaben zu den in der Lehrveranstaltung behandelten Themenbereichen. Die Punktzahl ist bei jeder Aufgabe mit angegeben.

| **Aufgabe** | **Mögliche Punkte** | **Erreichte Punkte** |
|---|---|---|
| A1 | 9 | |
| A2 | 6 | |
| A3 | 6 | |
| A4 | 6 | |
| A5 | 8 | |
| A6 | 20 | |
| A7 | 10 | |
| A8 | 25 | |
| | | |
| | | |
| **Summe** | 90 | |
| Evtl. Bonus | | |

**Bitte dokumentieren Sie die Antworten zu allen Aufgaben in der entsprechenden Textbox auf dem Blatt oder der entsprechend benannten Datei (z.B. aufgabe1.txt oder aufgabe7.py) in dem Ordner auf der VM.**

## Aufgabe 1

1) Was ist der Unterschied zwischen einer deklarativen und einer imperativen Programmiersprache und welchen Vorteil bieten die zwei Kategorien? Nutzen Sie gerne Beispiele zur Erläuterung der Unterschiede. (2 Punkte)

2) Es gibt Probleme in der Informatik, die sich heutzutage selbst mit aktuellen Hochleistungsrechnern und unter Verwendung kluger Algorithmen nicht mit vertretbarem Rechenaufwand bzw. in vertretbarer Zeit präzise lösen lassen. Nennen Sie Anwendungsfälle aus dem täglichen Leben, bei denen die Informatik genau dieses „Defizit“ ausgenutzt hat. Was würde passieren, wenn dieses „Defizit“ nicht mehr existieren würde? (1 Punkt)

3) Kreuzen Sie an: Was ist eine Aufgabe im Rahmen des Problemraums und was ist eine Aufgabe im Rahmen des Lösungsraums? (3 Punkte)

| **Aufgabe** | **Problemraum** | **Lösungsraum** |
|---|---|---|
| Sammeln, Verstehen und Definieren der Benutzeranforderungen. | | |
| Identifikation potenzieller Risiken, Abhängigkeiten und Interessen von Stakeholdern, die das Projekt beeinflussen könnten | | |
| Implementierung und Testen gemäß Anforderungen. | | |
| Auswahl geeigneter Programmiersprachen, Frameworks und Tools, die zur Lösung des Problems eingesetzt werden. | | |
| Erstellen des Systemdesigns, einschließlich Architektur, Datenmodellierung und Schnittstellendesign. | | |
| Erstellung von Benutzerszenarien, Use-Case-Beschreibungen und funktionalen Spezifikationen. | | |

4) Beantworten Sie folgende Fragen zur Turing-Maschine: (3 Punkte)
   a. Aus welchen Komponenten besteht eine Turing Maschine?
   b. Was sind die grundlegenden Operationen, die eine Turing-Maschine ausführen kann?
   c. Welche *praktische* Bedeutung hat die Turing-Vollständigkeit für die Wahl der Programmiersprache?

## Aufgabe 2

Erstellen Sie folgendes Programm mit einem iterativen Ansatz: (6 Punkte)
Geben Sie das Produkt der ersten *n* Primzahlen aus.
Der Wert von *n* soll zunächst vom Nutzer angefragt werden. Die Primzahlen sollen im Rahmen des Programms mit einem iterativen Verfahren bestimmt und ausgegeben werden.
Abschließend soll das Produkt der Primzahlen berechnet und ausgegeben werden.

Zur Erinnerung: Eine Primzahl ist eine Zahl, die genau zwei Teiler hat: 1 und sich selbst. Achtung: die Zahl 1 ist keine Primzahl, da sie nur einen Teiler hat.

$$\mathbb{P} = (2,3,5,7,11,13,17, \dots)$$

Beispielausgabe des Programms:

```
Geben Sie die Anzahl der Primzahlen ein, deren Produkt berechnet werden soll: 4
Primzahl 1: 2
Primzahl 2: 3
Primzahl 3: 5
Primzahl 4: 7

Produkt der ersten 4 Primzahlen: 210
```

## Aufgabe 3

Schreiben Sie einen einfachen Vokabeltrainer für eine vorgegebene Liste an Wörtern (als Dictionary). (6 Punkte)

- Das Programm soll dem Nutzer hintereinander fünf zufällige Wörter in Deutsch vorschlagen und um die englische Übersetzung bitten und das eingegebene Ergebnis prüfen.
- Bei fehlerhaften Eingaben soll ein entsprechender Hinweis erfolgen und eine neue Aufgabe gestellt werden (keine Wiederholung derselben Aufgabe)
- Die Dokumentation der richtigen Ergebnisse soll durch einen einfachen Fortschrittsbalken zu Beginn der Zeile dokumentiert werden, bei dem korrekt gerechnete Ergebnisse mit einem „X“ markiert werden und noch zu rechnende Aufgaben mit „_“.

**Hilfestellung**:
Nutzen Sie die Datei `gdp_ss23/src/aufgabe03.py` als Ausgangspunkt Ihrer Implementierung.
Nutzen Sie die Funktion `random.choice(list(vokabeln.keys()))`, um ein zufälliges deutsches Wort (als key) aus dem Dictionary auszuwählen.

Wörter dürfen mehrfach in einer Runde abgefragt werden (siehe erste Beispielausgabe).

**gdp_ss23/src/aufgabe03.py**

```python
# Einbinden der Bibliothek
import random

vokabeln = {
    "hallo": "hello",
    "bitte": "please",
    "ja": "yes",
    "nein": "no",
    "hund": "dog",
    "katze": "cat",
    "essen": "eat",
    "trinken": "drink",
    "schlafen": "sleep",
}

# Wählt ein zufälliges Wort aus der Vokabelliste aus
key = random.choice(list(vokabeln.keys()))

# Ausgabe des Wortpaares
print("Deutsch: ", key)
print("Englisch: ", vokabeln[key])
```

Im Folgenden sehen Sie zwei Beispielausgaben, die durch das Programm erzeugt werden könnten. Achtung: die gelb markierten Zahlen sind jeweils Nutzereingaben!

**Beispielausgabe 1** ohne Fehler in der Eingabe (Nutzereingaben in gelb):

```
Übersetze die folgenden fünf Wörter:
[_____] bitte: please
[X____] essen: eat
[XX___] nein: no
[XXX__] trinken: drink
[XXXX_] essen: eat
[XXXXX] Gratulation! Gut gemacht!
```

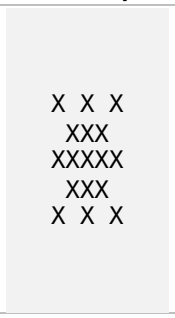
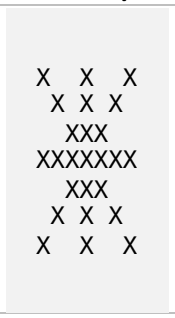
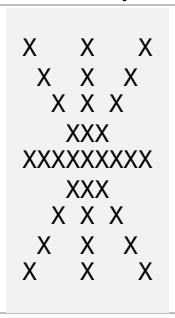
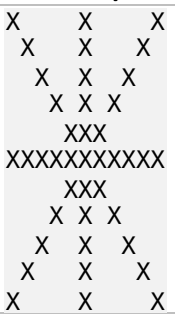
**Beispielausgabe 2** mit Fehlern in der Eingabe (Nutzereingaben in gelb):

```
Übersetze die folgenden fünf Wörter:
[_____] essen: eat
[X____] hallo: eat
Falsch, richtige Antwort wäre gewesen: hello
[X____] bitte: please
[XX___] bitte: please
[XXX__] hallo: hello
[XXXX_] essen: eat
[XXXXX] Gratulation! Gut gemacht!
```

## Aufgabe 4

Geben Sie auf der Kommandozeile einen vereinfachten Stern aus. Hierfür wird der Nutzer um Angabe der maximalen Breite/Höhe (B/H) des Sterns gebeten. Die minimale Breite beträgt 5. Zur Vereinfachung dürfen Sie davon ausgehen, dass nur ungerade Zahlen vorgegeben werden. (6 Punkte)

Im Folgenden drei Beispielausgaben für unterschiedliche Breiten/Höhen (B/H):

| Stern mit B/H 5: | Stern mit B/H 7: | Stern mit B/H 9: | Stern mit B/H 11: |
|---|---|---|---|
|  |  |  |  |









## Aufgabe 5

Erstellen Sie vier Funktionen, die für eine gegebene Kantenlänge *k*

- die Fläche des Quadrats [$A(k) = k^2$ ] mit Funktionsnamen `flaeche`,
- den Umfang des Quadrats [$U(k) = 4\,k$ ] mit Funktionsnamen `umfang`,
- die Oberfläche des Würfels [$O(k) = 4\,A(k)$] mit Funktionsnamen `oberflaeche`,
- das Volumen des Würfels [$V(k) = k^3$] mit Funktionsnamen volumen

zurückgibt. [8 Punkte]

Erstellen Sie weiterhin eine einfache Nutzerabfrage und geben alle berechneten Werte aus.

**Beispielausgabe** (Nutzereingaben in gelb):

```
Welche Kantenlänge hat das Quadrat bzw. der Würfel in cm? 5
Flaeche Quadrat: 25 cm^2
Umfang Quadrat: 20 cm
Oberfläche Würfel: 150 cm^2
Volumen Würfel: 125 cm^3
```

## Aufgabe 6

Die Oberfläche einer Kugel ergibt sich aus dem Radius durch folgende Gleichung:

$$O = 4\pi\, r^2$$

Der Nutzer möchte nun zu einer gegebenen Oberfläche *O* den zugehörigen Radius r bestimmen. Hierbei soll in der Lösung jedoch auf die Verwendung des Wurzel-Operators in Python verzichtet werden. Schreiben Sie daher bitte folgende zwei Funktionen:

1) Bestimmung des Radius *r* zu einer Oberfläche *O* mit Hilfe der linearen Suche (8 Punkte)
2) Bestimmung des Radius *r* zu einer Oberfläche *O* mit Hilfe der binären Suche (10 Punkte)

Erstellen Sie einen einfachen Benchmark: Wie viele Iterationen benötigen Ihre zwei Implementierungen? (2 Punkte)

## Aufgabe 7

Die Fibonacci-Reihe ist eine Reihe von Zahlen, bei der sich der Wert der aktuellen Zahl (ab n>2) aus der Summe der beiden vorherigen Zahlen ergibt:

$$f_n = f_{n-1} + f_{n-2} \text{ für } n > 2$$

Anfangswerte: $f_0 = 0, \qquad f_1 = f_2 = 1$

Die Sequenz sieht daher folgendermaßen aus: 0, 1, 1, 2, 3, 5, 8, 13, ...

Implementieren Sie nun die Fibonacci-Funktion `fib(n)`, die die n-te Zahl in der Fibonacci-Reihe zurückgibt. Nutzen Sie hierfür einen *rekursiven* Ansatz. (10 Punkte)

**Erwartete Ergebnisse:**

```
fib(0) = 0
fib(1) = 1
fib(2) = 1
fib(3) = 2
fib(4) = 3
fib(5) = 5
```

## Aufgabe 8

Ein Reisebüro bietet Reisen an. Reisen haben eine eindeutige ID, ein Reiseziel und einen Preis (pro Tag und Person). Es gibt unterschiedliche Typen von Reisen. So bietet das Reisebüro aktuell Party-, Kultur- und Fahrrad-Reisen an. Weiterhin hat das Reisebüro Kunden und Mitarbeiter. Beide sind Personen und haben einen Namen, eine Identifikationsnummer sowie eine Telefonnummer. Mitarbeiter haben zusätzlich ein Gehalt. Das Reisebüro ermöglicht es Kunden, Reisen zu buchen. Eine Buchung hat einen Start- und einen End-Zeitpunkt sowie einen Preis und bezieht sich auf ein Reiseziel.

1) Modellieren Sie die o.g. Zusammenhänge als UML-Modell (10 Punkte)

2) Implementieren Sie die grobe *Struktur* der Klassen Ihres UML-Modells (10 Punkte)
3) Implementieren Sie folgende zwei Funktionen und führen diese beispielhaft aus: (5 Punkte)
    a. Anbieten einer Reise im Reisebüro
    b. Buchung einer Reise eines Kundens durch einen Mitarbeiter

**Viel Erfolg bei der Bearbeitung der Aufgaben!**