Name: Vorname:

Geburtsdatum: Matrikelnummer:

Platz:

# Klausur
# Informatische Werkzeuge in den Geistes- und Sozialwissenschaften 1

13. Februar 2025

| | To be used for grading, do not write here | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| prob. | 1.1 | 1.2 | 1.3 | 2.1 | 2.2 | 3.1 | 3.2 | 3.3 | 4.1 | 4.2 | 4.3 | 4.4 | Sum | grade |
| total | 4 | 2 | 4 | 6 | 4 | 4 | 4 | 8 | 4 | 4 | 8 | 8 | 60 | |
| reached | | | | | | | | | | | | | | |

Die „Lösungen“ der Aufgaben in diesem Dokument sollen den Studierenden als Anfangspunkt für die Beantwortung der Aufgaben dienen. Trotz aller Bemühungen kann es zu Unvollständigkeiten oder sogar Fehlern in den „Lösungen“ kommen. Da die Korrektur und Benotung der gestellten Aufgaben niemals lediglich auf einen „Vergleich mit der Musterlösung“ hinausläuft, ist dies auch nicht so schlimm. In jedem Fall sollten die Studierenden die Lösungen nachvollziehen und im Prinzip mittels des Lehrstoffs selbst verifizieren können.

Sollten Sie „Lösungen“ finden, die Sie nicht verstehen oder sogar für fehlerhaft halten diskutieren Sie diese am besten mit den Tutor*innen oder auf dem Kursforum und benachrichtigen Sie die Lehrenden; wir werden sie dann gegebenenfalls baldigst korrigieren.

# 1 Grundlagen und Verständnis

**Aufgabe 1.1 (Bytes)** 4 Punkte
4 Min

Erklären Sie die Begriffe Bit, Byte, Kilobyte und Mebibyte. Erläuteren Sie außerdem kurz die Beziehungen, die diese Begriffe untereinander haben.

---

*Lösung:* Ein Bit ist die fundamentale Einheit für Information. Bits können entweder den Wert 1 oder 0 annehmen.

Ein Byte sind 8 Bit. Ein Kilobyte sind (Nach SI-Präfix-Nomenklatur) 1000 Byte. Manchmal werden auch 1024 Byte als ein „Kilobyte“ bezeichnet. Um hier Präzision in der Sprache zu erlauben wurde der Begriff „Kibibyte“ für 1024 Bytes eingeführt. Ein Mebibyte sind 1024 Kibibyte.

---

---

*Bewertung:* Es wird jeweils ein halber Punkt für jede korrekt erklärte Größte vergeben. Die letzten zwei Punkte kommen durch korrekte Gegenüberstellung / Begründung der beiden Präfixtraditionen (1000 vs 1024).

AC1 Bit Korrekt
Punkte: +0.5
Rückmeldung: Erklärt korrekt die Größe Bit

AC2 Byte Korrekt
Punkte: +0.5
Rückmeldung: Erklärt korrekt die Größe Byte

AC3 Kilobyte Korrekt
Punkte: +0.5
Rückmeldung: Erklärt korrekt die Größe Kilobyte

AC4 Mebibyte Korrekt
Punkte: +0.5
Rückmeldung: Erklärt korrekt die Größe Mebibyte

AC5 Begründung
Punkte: +2.0
Rückmeldung: Erklärt korrekt, warum die verschiedenen Prafixtraditionen existieren.

---

**Aufgabe 1.2 (Listen-Adressierung)** 2 Punkte
2 Min

In der Programmiersprache Python kann das erste Element einer Liste mit `liste[0]` angesprochen werden und das letzte Element mit `liste[-1]`.

Warum wird hier „von vorne“ bei 0 angefangen und „von hinten“ stattdessen mit 1?

---

*Lösung:* In Python gilt, dass `0 == -0`. Wenn also auch “rückwärts” 0-Indizierung vewendet werden würde, dann wäre es nicht möglich, das letzte Element einer Liste anzusteuern, weil `liste[-0] == liste[0]` wieder das erste Element wäre.

---

---

*Bewertung:* Die große Einsicht, die hier wichtig ist, ist `0 == -0` mit der Folge, dass die Alternative zum tatsächlichen Verhalten nicht eindeutig wäre. Wenn nur der zweite Teil genannt wird, ohne zu sagen, warum, gibt es einen Punkt Abzug.

AC1 Technische Begründung fehlt
Punkte: -1.0
Rückmeldung: Es wird nicht angegeben, dass 0 == -0.

**Aufgabe 1.3 (Cookies)** 4 Punkte
4 Min

Erklären Sie kurz das Konzept von sogenannten *Cookies* in der Webprogrammierung. Was ist ein Cookie? Wo werden sie angelegt? Wer kann darauf zugreifen? Nennen Sie außerdem ein Beispiel wofür ein Cookie verwendet werden könnte.

*Lösung:* Ein Cookie ist eine verhältnismäßig kleine Menge von Text, die clientseitig (d.h. auf der Festplatte der Nutzenden eines Services, also zum Beispiel einer Website) angelegt und gespeichert wird. In diesem Text werden Informationen gespeichert, die für den Betrieb des Services relevant sind (z.B. das Datum des letzten LogIns oder der Inhalt eines Warenkorbes).

Selbstverständlich hat der Service selbst Zugriff auf diese Informationen, allerdings haben das auch die Nutzenden selbst (!) und potentiell auch andere Services im Internet (!!) weswegen der Inhalt oft verschlüsselt ist.

*Bewertung:* Je einen Punkt für korrekte Beantwortung einer der gegebenen Fragen (Was? Wo? Wer? und Beispiel).

AC1 Was? (korrekt)
Punkte: +1.0
Rückmeldung: Abgabe erklärt korrekt, was ein Cookie ist.

AC2 Wo? (korrekt)
Punkte: +1.0
Rückmeldung: Abgabe erklärt korrekt, wo ein Cookie angelegt wird.

AC3 Wer? (korrekt)
Punkte: +1.0
Rückmeldung: Abgabe erklärt korrekt, wer auf Cookies zugreifen darf.

AC4 Beispel (korrekt)
Punkte: +1.0
Rückmeldung: Abgabe gibt ein korrektes Beispiel für Cookies.

# 2 Reguläre Ausdrücke

**Aufgabe 2.1 (Matchen Gegebener Regexes)** 6 Punkte
6 Min

Geben Sie einen String an, der gegen alle folgende regulären Ausdrücke matcht.

- `^.ei.e...[aeiou]i$`
- `^.{3}ß.{4}[^eiou].$`
- `^W.{6}H.{2}$`
- `^.{5}r\s.{3}$`

*Hinweis:* Gehen Sie methodisch vor! Alle regulären Ausdrücke matchen den gesamten String vom Anfang bis zum Ende und verraten Informationen über bestimmte Teile (z.B. konkrete Buchstaben an bestimmten Stellen). Nur ein einziger String matcht alle vier Ausdrücke.

*Lösung:* Der Lösungsstring ist `"Weißer Hai"`.

*Bewertung:* 1.5 Punkte für jeden regulären Ausdruck, den der gegebenen String erfüllt.

AC1 Regex1
Punkte: +1.5
Rückmeldung: Stringt matcht den ersten Regex

AC2 Regex2
Punkte: +1.5
Rückmeldung: Stringt matcht den zweiten Regex

AC3 Regex3
Punkte: +1.5
Rückmeldung: Stringt matcht den dritten Regex

AC4 Regex4
Punkte: +1.5
Rückmeldung: Stringt matcht den vierten Regex

**Aufgabe 2.2 (Umgehung von Zensur)** 4 Punkte
4 Min

Nach der Machtergreifung eines autoritären aber glücklicherweise informatisch inkompetenten Regimes werden alle Kommunikationen mittels regulärer Ausdrücke zensiert. Es ist ihre Aufgabe, diese Zensur zu umgehen.

Die Nachricht die Sie heute dringend übermitteln wollen ist diese Telefonnummer[1]:

"0800 220 5555"

Allerdings wissen Sie, dass auf allen Nachrichten folgende Substitution ausgeführt wird, was die direkte Übermittlung unmöglich macht: `message = re.sub("\d+", "", message)`

Geben Sie einen String an, aus dem ein ander Mensch die Telefonnummer auslesen kann, auch nachdem die obige RegEx-Substitution durchgeführt wurde.

*Lösung:* Hier ist eine mögliche Lösung:

`"Null Acht Null Null Zwei Zwei Null Fünf Fünf Fünf Fünf"`

*Bewertung:* Jede Umschreibung der einzelnen Zahlen oder der Zahl als ganzes (dann muss jedoch die führende Null separat behandelt werden) gilt als korrekte Lösung, solange sie keine Ziffern benutzt.

AC1 Gibt einen Regex
Punkte: 0.0
Rückmeldung: Abgabe enthält einen Regex statt einen Fließtext.

AC2 Benutzt Ziffern.
Punkte: 0.0
Rückmeldung: Abgabe benutzt (nur) Ziffern, die ersetzt würden.

[1] Im echten Leben ist dies die Hotline von MausLive, ein Radioformat der Sendung mit der Maus

# 3 Digitale Dokumente

**Aufgabe 3.1 (Cascading Style Sheets)** 4 Punkte
4 Min

Geben Sie ein Beispiel für zwei (unterschiedliche) *korrekte* und zwei (unterschiedliche) *inkorrekte* CSS-Regeln.

Erklären Sie für die letzten beiden Regeln, warum genau diese kein korrektes CSS sind und was verändert werden müsste, damit sie das wären.

---

*Lösung:* Beispiele für korrekte CSS Regeln:

- h1 { text-align: center; }
- body { background-color: #FF9900; }

Beispiele für inorrekte CSS Regeln:

- p { font-family: verdana font-size: 14pt }
  (Keine Trennung durch Semikola)
- h1 [ text-align: ]
  (Falsche Klammern, kein Wert zugeordnet)

---

---

*Bewertung:* Je ein halber Punkt für die tatsächlichen Regeln und je ein Punkt für korrekte Erklärungen, was an einer angegebenen Regel nicht stimmt.

AC1 Korrekte richtige Regel
Punkte: +0.5
Rückmeldung: Abgabe einer korrekterweise richtigen CSS-Regel

AC2 Korrekte falsche Regel
Punkte: +0.5
Rückmeldung: Abgabe einer korrekterweise falschen CSS-Regel

AC3 Erklärung falsche Regel
Punkte: +1.0
Rückmeldung: Abgabe einer korrekten Erklärung einer falschen Regel.

---

**Aufgabe 3.2 (URI zu XML)** 4 Punkte
4 Min

Gegeben ist die folgende URI:

```
https://jupyter.kwarc.info:64042/greet/Totoro?colour=red#sparks
```

Diese URI soll nun zur Weiterverarbeitung in XML umgewandelt werden.

Geben Sie einen gültigen XML-Baum mit den fünf Bestandteilen einer URI (scheme, authority, path, query, fragment) und den jeweiligen Inhalten an. Wählen Sie dabei passende Namen für die Knoten und bedenken Sie, dass alle diese Bestandteile auch in XML zu einer URI zusammen gefasst werden sollen. Trennzeichen, die *nicht* zu einem bestimmten Bestandteil gehören, sollen dabei wegfallen.

---

*Lösung:*

```
<uri>
    <scheme>
        https
    </scheme>
    <authority>
        jupyter.kwarc.info:64042
    </authority>
    <path>
        /greet/Totoro
    </path>
    <query>
        colour=red
    </query>
    <fragment>
        sparks
    </fragment>
</uri>
```

---

*Bewertung:* Vergeben wird 1.5 Punkte für die generelle Struktur (URI-Knoten mit den einzelnen Bestandteilen als Kindern) und jeweils 0.5 Punkte für korrekt zugeordnete Informationen zu den Bestandteilen.

Abzüge gibt es z.B. für überflüssige Trennzeichen (jeweils 0.5 Punkte) oder für das Fehlen von einem übergeordneten Knoten für URI (0.5 Punkte).

AC1 Korrekte Struktur
Punkte: +1.5
Rückmeldung: Generelle XML Struktur wurde korrekt angegeben.

AC2 Scheme korrekt
Punkte: +0.5
Rückmeldung: Der Knoten für Scheme hat den korrekten Inhalt.

AC3 Authority korrekt
Punkte: +0.5
Rückmeldung: Der Knoten für Authority hat den korrekten Inhalt.

AC4 Path korrekt
Punkte: +0.5
Rückmeldung: Der Knoten für Path hat den korrekten Inhalt.

AC5 Query korrekt
Punkte: +0.5
Rückmeldung: Der Knoten für Query hat den korrekten Inhalt.

AC6 Fragment korrekt
Punkte: +0.5
Rückmeldung: Der Knoten für Fragment hat den korrekten Inhalt.

AC7 Überflüssige Trennzeichen
Punkte: -0.5
Rückmeldung: Trennzeichen zwischen Bestandteilen vorhanden.

AC8 URI-Knoten fehlt
Punkte: -0.5
Rückmeldung: Kein übergeordneter Knoten für URI.

---

**Aufgabe 3.3 (Lieblingslied Template)** 8 Punkte
8 Min

In dieser Aufgabe sollen Sie ein `bottle` (bzw. `STPL`) *Template* schreiben, dass (je nach Umstand, s.u.) entweder genau ein Lied oder eine Liste von Liedern anzeigt.

Sie wissen, dass das Template eine Variable `highlight` und eine Variable `songs` übergeben bekommt. Erstere ist ein Boolean, letztere eine Liste von Strings.

Falls die Variable `highlight` `True` sein sollte oder wenn in der Liste `songs` nur ein einziges Element enthalten ist, dann soll ihr Template am Ende eine Überschrift "Mein Lieblingslied" haben und in einem Text darunter den Titel erwähnen.

In allen anderen Fällen soll die Überschrift "Meine Lieblingslieder" lauten, gefolgt von einer HTML-Liste (nummeriert oder nicht) aller Elemente von `songs`.

---

*Hinweis:*

- Gefragt ist nur das Template, Sie brauchen *keinen* Code für den Server.
- Ordentlich aufgerufen soll ihr Template eine komplette (gültige) HTML-Seite zurück geben.

---

---

*Lösung:* Hier ist eine mögliche Lösung:

```
<html>
    <body>
    % if highlight or (len(songs) == 1):
    <h1>Mein Lieblingslied</h1>
    Mein absolutes Lieblingslied ist {{songs[0]}}!
    % else:
    <h1>Meine Lieblingslieder</h1>
    Meine Lieblingslieder sind:
    <ul>
    % for item in songs:
      <li>{{item}}</li>
    % end
    </ul>
    % end
    </body>
</html>
```

---

# 4 Programmieren in Python

**Aufgabe 4.1 (Passfile Reverse-Engineering)** 4 Punkte

4 Min

Gegeben ist folgende Python-Funktion:

```
def check_passfile():
    with open('passfile.txt') as pfile:
        text = pfile.read()

    dic = {}
    with open('passfile.txt') as pfile:
        for line in pfile.readlines():
            if line.startswith('auto'):
                dic[line] = len(line)

    pass1 = len(text) % 2 == 0
    pass2 = len(dic.keys()) == 3

    return pass1 and pass2
```

Geben Sie einen möglichen Inhalt der Datei `passfile.txt` an, sodass die Funktion `check_passfile()` den Wert `True` zurück gibt.

---

*Lösung:* Hier ist eine mögliche Lösung:

```
auto
automatic
this line has 50 characters including the newline
automaton
```

In diesem Beispielcode werden zwei verschiedene Bedingungen mit dem gleichen Quelltext überprüft. Einerseits sollen genau drei (verschiedene, Doppelungen werden ignoriert) Zeilen mit der Präfix `"auto"` beginnen. Andererseits soll die Anzahl der Zeichen in der Datei insgesamt gerade sein. Dabei ist es wichtig, die Zeichen für Zeilenumbrüche nicht zu vergessen. Im obigen Beispiel könnte die dritte Zeile auch weg gelassen werden, weil sie eine gerade Anzahl an Zeichen hat.

---

---

*Bewertung:* Je zwei Punkte für das korrekte angeben von drei Begriffen mit der korrekten Präfix (“auto”) und der insgesamt geraden Anzahl von Zeichen. Nicht-offensichtlich Whitespace-Zeichen müssen dabei gekennzeichnet werden. Je einen Punkt Abzug falls entweder Newline-Zeichen nicht mit bedacht wurden oder es (fehlerhafte) Doppellungen in den Begriffen gibt.

AC1 Drei Auto-Begriffe
Punkte: +2.0
Rückmeldung: Abgabe nennt drei Begriffe mit der auto-Präfix.

AC2 Gerade Anzahl
Punkte: +2.0
Rückmeldung: Abgabe gibt eine Datei mit gerader Anzahl Zeichen an.

AC3 Newlines missachtet
Punkte: -1.0
Rückmeldung: Abgabe ignoriert bei der Länge die (impliziten) Newline-Zeichen.

AC4 Doppellungen
Punkte: -1.0
Rückmeldung: Bei den genannten Begriffen gibt es Doppelungen, die zu Fehlern führen.

---

**Aufgabe 4.2 (Mentales Python)** 4 Punkte
4 Min

Gegeben ist folgendes Python-Programm:

```
# What will this code print to console?

my_list = []

for x in range(5):
    for y in range(3):
        my_list.append(x)

my_second_list = []

i = 0
while i < len(my_list):
    if i % 3 == 0:
        my_second_list.append(my_list[i])
    i = i + 1

print(my_list)
print(my_second_list)
```

Geben Sie an, was dieses Programm ausgeben („printen“) wird, wenn es ausgeführt wird.

---

*Lösung:* Die äußere for-Schleife läuft über `range(5)`, also von 0 bis 4, während die innere for-Schleife jeweils drei Mal den aktuellen Wert der äußeren Laufvariable (`x`) der Liste `my_list` hinzufügt.

Die while-Schleife benutzt eine "manuelle"Laufvariable, die über die gesamte Länge von `my_list` läuft, aber nur das aktuelle Element in die zweite Liste (`my_second_list`) überträgt, falls der aktuelle Wert der Laufvariable ohne Rest durch 3 teilbar ist. Das bedeutet, dass nur jedes dritte Element tatsächlich übertragen wird.

Die korrekte Lösung hier ist also `"[0,0,0,1,1,1,2,2,2,3,3,3,4,4,4]"` für `my_list` und `"[0,1,2,3,4]"` für `my_second_list` (Strings weil ultimativ alles was geprinted werden kann ein String ist, `print()` übernimmt dabei die Umwandlung), wir akzeptieren aber selbstverständlich auch `[0,0,0,1,1,1,2,2,2,3,3,3,4,4,4]` und `[0,1,2,3,4]`.

---

---

*Bewertung:* Je zwei Punkte für einen korrekten Inhalt einer ausgegebenen Variable. Einen Punkt Abzug gibt es, wenn zwar der Inhalt prinzipiell richtig erkannt wurde, aber Unsauberkeiten in der genauen Ausgabe existieren (z.B. vergessene Listen-Klammern / Kommata oder beide Ausgaben ohne Newline).

AC1 Erste Liste richtig.
Punkte: +2.0
Rückmeldung: Abgabe identifiziert Inhalt von `my_list` korrekt.

AC2 Zweite Liste richtig.
Punkte: +2.0
Rückmeldung: Abgabe identifiziert Inhalt von `my_second_list` korrekt.

AC3 Unsauber
Punkte: -1.0
Rückmeldung: Abgabe enthält Unsauberkeiten bzgl. Python

---

**Aufgabe 4.3 (Webserver-Route zum Finden des Maximums)** 8 Punkte
8 Min

Geben Sie eine vollständige (!) Route für einen `bottle`-Webserver an, mit der das Maximum (d.h. der größte Wert) von drei natürlichen Zahlen ermittelt werden kann, welche als Wildcard-Parameter übergeben werden.

Würde Ihre Route zum Beispiel mit `/maximum/17/0/4` aufgerufen, so sollte `"17"` zurück gegeben werden. Beim Aufruf `/maximum/2/2/2` sollte `"2"` zurück gegeben werden.

Eventuelle Eingabefehler müssen Sie nicht abfangen. Ebenfalls müssen Sie sich nicht um die Einbettung in valides HTML kümmern. Geben Sie einfach die entsprechende Antwort als String zurück.

---

*Lösung:* Hier ist eine mögliche Lösung:

```
@route("/maximum/<a>/<b>/<c>")
def iwgs_maximum(a,b,c):
    """Gibt das Maximum der drei Werte zurueck."""

    # Konvertierung zu Integer ist nicht optional,
    # da sonst alphanumerisch verglichen wird.
    a = int(a)
    b = int(b)
    c = int(c)

    if (a >= b) and (a >= c):
        largest = a
```

```
    elif (b >= a) and (b >= c):
        largest = b
    else:
        largest = c

    return str(largest)
```

---

**Aufgabe 4.4 (Webserver-Route zum Entfernen von Duplikaten)** 8 Punkte

8 Min

Geben Sie eine vollständige (!) Route für einen `bottle`-Webserver an, mit der Duplikate aus einer Liste entfernt werden können. Die Liste wird als String im Wildcard-Parameter übergeben, und einzelne Elemente der Liste sind durch Bindestriche (`"-"`) getrennt (s.u.).

Würde Ihre Route also zum Beispiel mit `/remove-duplicates/1-foo-1-5-5-10-1` aufgerufen, so sollte `"1-foo-5-10"` zurück gegeben werden.

Eventuelle Eingabefehler müssen Sie nicht abfangen. Ebenfalls müssen Sie sich nicht um die Einbettung in valides HTML kümmern. Geben Sie einfach die entsprechende Antwort als String zurück.

---

*Lösung:* Hier ist eine mögliche Lösung:

```
@route("/remove-duplicates/<mystring>")
def iwgs_remove_duplicates(mystring):
    """Entfernt Duplikate aus dem String"""

    newList = []
    for elem in mystring.split("-"):
        if elem not in newList:
            newList.append(elem)

    output = newList[0]
    for elem in newList[1:]:
        output = f"{output}-{elem}"

    return output
```

---