Abschlussklausur zu Programmierung 1 im Wintersemester 2012/13

Geschrieben am 08.02.2013

MUSTERLÖSUNG

Vorname _____

Nachname _____

Matrikelnummer _____

Geburtsdatum _____

Studiengang _____

Klausur für 9 Credit Points werten? Ja ☐

Vom Prüfer auszufüllen:

Aufgabe	1	2	3	4	5	6	7	8	9	10	Σ
Punkte	10	10	10	10	10	10	10	10	10	10	100
Davon erreicht											

Klausurpunkte _____

\+ Bonuspunkte _____

Σ _____

Note _____

Aufgabe 1: Multiple Choice — Punkte: 10/ __

Eine Frage gilt dann als korrekt beantwortet, wenn alle richtigen und keine falschen Antworten angekreuzt sind. Mindestens eine Antwort ist immer richtig.

PRG 1.4
Skript V1
Folien V1

(a) (1 Punkt) **Algorithmus**
Algorithmen, die zu jedem Zeitpunkt der Ausführung maximal eine Möglichkeit der Programmfortsetzung besitzen, sind:

- ☐ Determiniert
- ☐ Determinante
- ☐ Determination
- ☒ **Deterministisch**

PRG 5.5
EPR 4.3a
Skript V9
Folien V9

(b) (1 Punkt) **Formale Sprachen**
Gegeben sei die folgende Grammatik in EBNF mit Startsymbol `S`.

```
S = {P} | {M};
P = P, "+", {M} | Z, {Z};
M = M, "*", M | Z, {Z};
Z = "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9";
```

Welche Wörter gehören zu der durch die Grammatik definierten Sprache?

- ☐ 10*20
- ☒ **64+32+**
- ☐ P+MMM
- ☒ **128+64*32**

PRG 1.3
Skript V1
Folien V1

(c) (1 Punkt) **Rechner-Architektur**
Welche der folgende Komponenten gehören zur von-Neumann-Architektur?

- ☐ Lesewerk
- ☐ Hauptwerk
- ☒ **Rechenwerk**
- ☒ **Speicherwerk**

Folien V3
Skript V3/4

(d) (1 Punkt) **Pythonanalyse**
Was berechnet der folgende Python-Code?

```
1 x, y = 7, 21
2 x += x // y * 3.1
3 print(x)
```

- ☐ `7`
- ☐ `8`
- ☒ `7.0`
- ☐ `NameError: name is not defined`

Folien V3
Skript V3/4

(e) (1 Punkt) **Python Ausdrücke**
Welche der folgenden Ausdrücke werden von Python (Version 3.x) zu `True` ausgewertet?

- ☒ **`bool(12 % 5.0)`**
- ☐ `bool(1 >> 2 >> 3)`
- ☐ `bool(1.0 == 0 and 0.0 != 1.0)`
- ☒ **`bool(not True or 12 * 12 / 12)`**

Folien V8
Skript V8

(f) (1 Punkt) **Built-in Datentypen**
Welche der folgenden Python-Builtins sind veränderlich (*mutable*)?

- ☒ **`Set`**
- ☐ `Tuple`
- ☐ `String`
- ☒ **`Dictionaries`**

PRG 5.1/2
Folien V8
Skript V8

(g) (1 Punkt) **Listen**
Was gilt für Pythons Listen?

- ☐ Listen sind immer sortiert
- ☐ Listen dürfen nicht leer sein
- ☐ Listenelemente sind immer eindeutig
- ☒ **Listen können beliebige Elemente verwalten**

PRG 6.3
Folien V5
Skript V5/6

(h) (1 Punkt) **Funktionen & Prozeduren**
Welche Unterschiede gelten für Funktionen und Prozeduren in Python?

- ☐ Python trifft eine statische Unterscheidung
- ☒ **Python ergänzt Prozeduren automatisch zu Funktionen**
- ☐ Python macht eine Pseudounterscheidung mit Hilfe von `__`
- ☐ Python verwendet die Argumentenliste zur Unterscheidung

PRG 7.3

(i) (1 Punkt) **Objektorientierung**
Welche Relation besteht zwischen einer Klasse und ihrer Basisklasse?

- ☐ Komposition
- ☐ Spezifikation
- ☐ Lokalisierung
- ☒ **Generalisierung**

Folien 11
Skript 11

(j) (1 Punkt) **Prozess**
Woraus besteht ein *Prozess* unter anderem?

- ☒ **Stack**
- ☐ Scheduler
- ☒ **Prozesskontext**
- ☒ **Programmdaten**

Matrikelnummer _____

Aufgabe 2: Zahlendarstellung

Punkte: 10/ __

PRG 2.2b
Folien V3
Skript V3/4

(a) Hexadezimal ⇔ Dual

1. (1 Punkt) Konvertieren Sie die Dualzahl $1110\ 0011\ 0101_2$ zur Basis 16.

1. $E35_{16}$

2. (1 Punkt) Konvertieren Sie die Hexadezimalzahl $9D7_{16}$ zur Basis 2.

2. $1001\ 1101\ 0111_2$

Folien V3
Skript V3/4

(b) Dezimal ⇔ Zweierkomplement

1. (1 Punkt) Konvertieren Sie die Zahl $1000\ 0111_2$ des Zweierkomplementes mit einer Wortlänge von 8 Bit in eine vorzeichenbehaftete Dezimalzahl.

1. -121_{10}

2. (1 Punkt) Konvertieren Sie die vorzeichenbehaftete Dezimalzahl -123_{10} in das Zweierkomplement für eine Wortlänge von 8 Bit.

2. $1000\ 0101_2$

PRG 2.3a

(c) (2 Punkte) Gegeben sei das dem IEEE-754 entlehnte, reduzierte 8-Bit Format mit $s = 1$, $e = 4$ und $m = 3$ Bit. Ab welchem ganzzahligen Wert i ist es mit diesem Format nicht mehr möglich i, als 8-Bit Gleitkommazahl, exakt darzustellen? Wie kommen Sie zu diesem Ergebnis?

Lösung:

1 Punkte für die Begründung Der Exponent unterteilt die Zahlengerade in Intervalle der Länge 2^e. Die Mantisse unterteilt jedes Intervall wiederum in 2^m diskrete Werte. Ist die Länge des Intervalls echt größer als die Anzahl der diskrete Unterteilungen ist die exakte Darstellung von i nicht mehr möglich.
1 Punkte für die Antwort Also, wenn $i > 2^3 = 16$ gilt, kann nicht mehr jede Ganzzahl innerhalb des Formates abgebildet werden.

PRG 2.2
Folien V3
Skript V3/4

(d) (2 Punkte) Welche Dezimalzahl z wird mit folgendem Bitmuster gemäß *IEEE-754* Codiert? **Geben Sie den Rechenweg an.**

1 1 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

z = - 1 3 . 2 5

Lösung:

0,5 Punkte pro s, e, m und z

$$s = 1_2 \rightarrow \underline{\mathbf{1_{10}}} \tag{1}$$
$$e = 10000010_2 \rightarrow 130_{10} - 127_{10} = \underline{\mathbf{3_{10}}} \tag{2}$$
$$m = 10101000000000000000000_2 \rightarrow 1_{10} + \frac{1}{2}_{10} + \frac{1}{8}_{10} + \frac{1}{32}_{10} \tag{3}$$
$$= 1.0_{10} + 0.5_{10} + 0.125_{10} + 0.03125_{10} = \underline{\mathbf{1.65625_{10}}} \tag{4}$$
$$z = (-1_{10})^s \cdot 2^e \cdot m \tag{5}$$
$$= (-1_{10})^{-1} \cdot 2^3_{10} \cdot 1.65625_{10} \tag{6}$$
$$= -1_{10} \cdot 8_{10} \cdot 1.65625_{10} = \underline{\mathbf{-13.25_{10}}} \tag{7}$$

PRG 2.2
Folien V3
Skript V3/4

(e) (2 Punkte) Geben Sie das Bitmuster und den Rechenweg der Zahl $z = 5.75$ gemäß *IEEE-754* mit einfacher Genauigkeit an.

0 1 0 0 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Lösung:

0,5 Punkte pro s, e, m und Bit-Pattern, bei Rechenfehlern Dezimal $\rightarrow$ Binär einmalig $0,5$ Punkte Abzug.

$$s = \begin{cases} 1, & \text{wenn negativ,} \\ 0, & \text{wenn positiv} \end{cases}$$
$$\Rightarrow s = \mathbf{0}$$

$$e = \begin{cases} \lfloor ln_2(|z|) \rfloor & \text{wenn } z \geq 1 \\ \lfloor ln_2(|z|) \cdot |z| \rfloor & \text{wenn } z < 1 \end{cases}$$
$$\Rightarrow e = 2_{10} + 127_{10} = 129_{10} \rightarrow \mathbf{10000001_2}$$

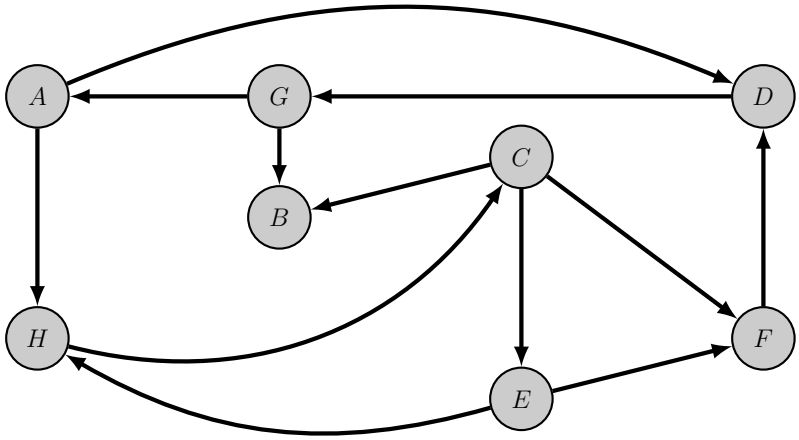
$$m = \frac{|z|}{2^e} = \frac{5.75_{10}}{4_{10}} = \mathbf{1.4375_{10}}$$
$$\Rightarrow 1.4375 = 1.0_{10} + 0.25_{10} + 0.125_{10} + 0.0625_{10} = 1 + \tfrac{1}{4} + \tfrac{1}{8} + \tfrac{1}{16}$$
$$\Rightarrow \mathbf{01110000000000000000000_2}$$

Aufgabe 3: Python-Datentypen **Punkte: 10/ __**

PRG 8.1
Folien V12
Skript V12

(a) (1 Punkt) Rekonstruieren Sie den gerichteten Graphen $\mathcal{G}$ anhand der Adjazenzliste `L` für die Knoten `A`, `B`, `C`, `D`, `E`, `F`, `G` und `H`.

```
L = [ [A, [H, D]],
      [B, []],
      [C, [B, E, F]],
      [D, [G]],
      [E, [H, F]],
      [F, [D]],
      [G, [A, B]],
      [H, [C]] ]
```



Lösung:

$$Punkte = \begin{cases} 0.0, & \textbf{0-6 Kanten richtig} \\ 0.5, & \textbf{7-9 Kanten richtig} \\ 1.0, & \textbf{10-12 Kanten richtig} \end{cases}$$

Ohne Pfeile 0,5 Abzug

PRG 8.1
Folien V12
Skript V12

(b) (1 Punkt) Geben Sie die Adjazenzmatrix $\mathcal{M}$ des Graphen $\mathcal{G}$ an[1].

$$\mathcal{M} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \\ G \\ H \end{array} \begin{array}{c} \begin{array}{cccccccc} A & B & C & D & E & F & G & H \end{array} \\ \begin{pmatrix} & & & 1 & & & & 1 \\ & & & & & & & \\ & 1 & & & 1 & 1 & & \\ & & & & & & 1 & \\ & & & & & 1 & & 1 \\ & & & 1 & & & & \\ 1 & 1 & & & & & & \\ & & 1 & & & & & \end{pmatrix} \end{array}$$

[1] Die Matrix muss nicht mit Nullen aufgefüllt werden.

(c) (2 Punkte) Gegeben sind die folgenden Python Codezeilen:

```
1 a = 2 * 3 ** 80 + 1
2 b = 2 << 42 // 15 % 3
3 c = 4 / (4 - 2 * 3)
4 d = 1 < 2 % 3 + 4
```

EPR 4
Handzettel
Folien V2
Skript V2

Tragen Sie die Operatoren in der Reihenfolge in die Kästchen ein, in der Sie ausgewertet werden.

Der Operator wird als	1.	2.	3.	ausgewertet.
Zeile 1	`**`	`*`	`+`	
Zeile 2	`//`	`%`	`<<`	
Zeile 3	`*`	`-`	`/`	
Zeile 4	`%`	`+`	`<`	

PRG 2.2
Handzettel
Folien V2
Skript V2

(d) (1 Punkt) Von welchem `type` sind die Variablen `c` und `d`?

1. `c`

1. **`float`**

2. `d`

2. **`bool`**

Handzettel
Skript V3
Handzettel

(e) (1 Punkt) Was sind die Operatoren für die folgenden Operationen in Python?

1. Bitweise Konjunktion

1. **`&`**

2. Logische Disjunktion

2. **`or`**

Folien V3
Skript V3/4

(f) (1 Punkt) Wie lauten die größte und die kleinste Zahl im Zweierkomplement, die sich mit einer Blockgröße von n-Bit darstellen lassen?

$$[-2^{n-1}, 2^{n-1} - 1]$$

Allgemeine Verständnisfrage

(g) (1 Punkt) Welchen Text gibt der nachstehende `print`-Befehl aus?

```
print("irQeelGaftPetspäfKw NedsDrAiVH"[-1:0:-2])
```

(g) **Hirse Käsetaler**

Folien V4
Skript V3/4

(h) (1 Punkt) Wie wird die implizite Typkonvertierung bezeichnet?

(h) **Coertion**

Folien V2
Skript V2

(i) (1 Punkt) Was ist der Vorteil der erweiterten Zuweisungen?
Die Variable muss nur einmal aufgelöst werden.

Aufgabe 4: Kontrollstrukturen **Punkte: 10/ __**

Allgemeines Verständnis

(a) (2 Punkte) Überführen Sie die Funktion $p(s,i) \to \mathbb{R}$ mit $s,i \in \mathbb{N}$ in äquivalenten Pythoncode.

$$p(s,i) = \prod_{n=1}^{i} \frac{1}{n \bmod s}$$

Lösung:

Wichtig ist die Klammerung in Zeile 4 (-1 Punkt)
Wichtig ist das +1 in Zeile 3 (-0.5 Punkt)

```
1 def p(s:int, i:int) -> float:
2     prod = 1.0
3     for n in range(1, i + 1):
4         prod *= 1 / (n % s)
5     return prod
```

Allgemeines Verständnis

(b) (1 Punkt) Überführen Sie die Funktion `f(x)` für die Folge `a` in mathematische Summen Schreibweise.

```
1 def f(x:int, a:list) -> float:
2     n = len(a)
3     return sum([a[i] * x ** i for i in range(0, n)])
```

Lösung:

$$f(x,a) = \sum_{i=0}^{n-1} a_i \cdot x^i, \text{ mit } n = |a|$$

Allgemeines Verständnis

(c) (1 Punkt) Welche Werte müssen `a, b` und `c` zugeordnet sein, damit das Programm „`-4, -2, 0, 2, 4, 6, 8, 10, `“ ausgibt?

```
1 def unknown(a, b, c):
2     while b < a:
3         print(b, end=", ")
4         b -= c
```

1. `a`
2. `b`
3. `c`

1. `11 bzw. 12`
2. `-4`
3. `-2`

Folien V5/6
Skript V5/6

(d) (1 Punkt) Über welche Schleifentypen (nicht Schlüsselwörter) verfügt Python?

Lösung:
Je 0,5 Punkte für die Schleifentypen

- Vorprüfende
- Zählschleife (Streng genommen ist `for/in` ein Iterator)

Folien V5/6
Skript V5/6

(e) (1 Punkt) Was entspricht in Python dem `switch-/case`-Statement?

Lösung:
Python kennt das `switch-/case`-Statement nicht. Das Verhalten kann jedoch mit dem `if-/else` bzw. `if-/elif-/else`-Statement nachgeahmt werden.

Folien V5/6
Skript V5/6

(f) (1 Punkt) Nach welchem Mechanismus findet die Parameterübergabe in Python statt?

Lösung:

Call by Object (Reference) oder *Call by Sharing*

EPR 1.1b
Folien V2
Skript V2

(g) (1 Punkt) Wie lautet das Schlüsselwort für die *leere Anweisung* in Python?

Lösung:

pass

Allgemeines Verständnis
Skript V5/6

(h) (2 Punkte) Welche Ausgabe liefert die `print`-Anweisung in Zeile 16?

```
1  x, y = 8, 4
2  def f(y):
3      global x
4      x = y
5      return x * 3
6
7  def g(y):
8      global x
9      if x >= 5:
10         return f(x // 3)
11     else:
12         y += 4
13         return x - y
14
15 for i in range(5):
16     print(g(i), end='->')
```

Lösung:

```
1 6->-3->-4->-5->-6->
```

0,5 Punkte Abzug wenn das -> am Ende fehlt

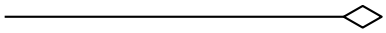
Aufgabe 5: OOP & OOAD **Punkte: 10/ __**

PRG 7.3b
Folien V11

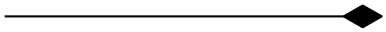
(a) (3 Punkte) Zeichnen Sie die UML-Symbole für die folgenden Elemente:

1. **Assoziation**

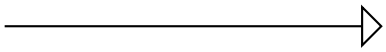
2. **Aggregation**



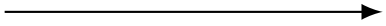
3. **Komposition**



4. **Generalisierung**



5. **Synchroner Operationsaufruf**



6. **Antwort des Synchronen Operationsaufrufes**

PRG 6.4c
Folien V10
Skript V10

(b) (1 Punkt) Mit welchen *besonderen* Methoden werden Objekte *im Allgemeinen* instantiiert, bzw. wird der von einem Objekt belegte Speicher freigegeben?

Lösung:

- Konstruktor
- Destruktor

EPR 5.2/5.3
Folien V10
Skript V10

(c) (2 Punkte) Erläutern Sie den Begriff *„Überschreiben“* im Kontext der objektorientierten Programmierung.

Lösung:

„Überdeckt“ ein neues Merkmal ein bei der Vererbung übernommenes Merkmal, dann spricht man von Überschreiben. Man kann also einzelne Methoden neu implementieren und außerdem eigene Methoden und Attribute hinzufügen.

(d) (2 Punkte) Was ist notwendig, um von *Polymorphie* im Kontext der objektorientierten Programmierung zu sprechen?

EPR 5.2/5.3
Folien V10
Skript V10

Lösung:

- Ein Klassenhierarchie,
- das Überschreiben von Merkmalen - einer - Basisklasse
- Das Überladen von Methoden zählt auch zur Polymorphie, braucht aber keine Klassenhierarchie.

(e) (1 Punkt) Wie viele Referenzen kann es in Python auf ein Objekt geben?

Transfer
Folien V10
Skript V10

Lösung:

Beliebig viele.
Nur durch den zur Verfügung stehenden Speicher begrenzt.

(f) (1 Punkt) Welche Sichtbarkeitsmodifikatoren gibt es in der UML und mit welchen Symbolen werden sie repräsentiert?

Folien V10
Skript V10

Lösung:

- public `+`
- private `-`
- package `~`
- protected `#`

Aufgabe 6: Testen

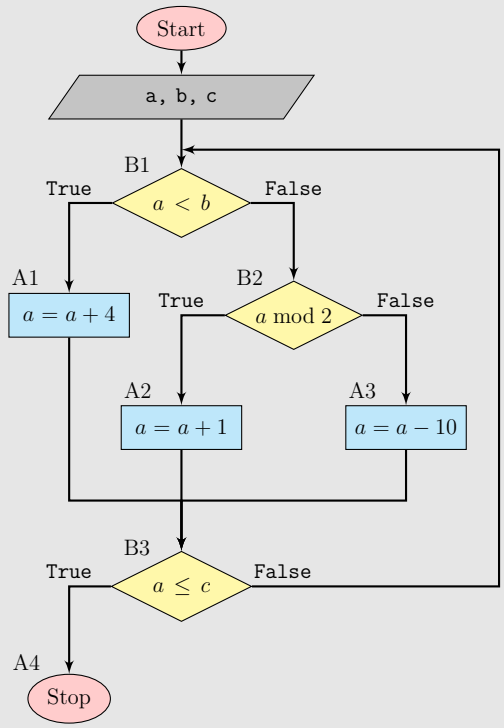
Punkte: 10/ __

PRG 3.3a/4.2b
Folien V5/6
Skript V5/6

(a) (1 Punkt) Übertragen Sie den nachstehenden Code in ein Flussdiagramm.

```
1  a, b, c = input("a, b, c")
2  while True:
3      if a < b:          ## B1
4          a += 4         ## A1
5      else:
6          if a % 2:      ## B2
7              a += 1     ## A2
8          else:
9              a -= 10    ## A3
10     if a <= c:         ## B3
11         break          ## A4
```

Lösung:



(b) (2 Punkte) Geben Sie Testfälle für eine Zweigüberdeckung des Codes aus Aufgabenteil *a* an. Benennen Sie die Reihenfolge, in der die Anweisungen und Bedingungen (`B1, B2, B3`) durchlaufen werden.

PRG 4.2
Folien V5/6
Skript V5/6

- Pfad: `B1, A1, B3, A4`
  für `a=-4`, `b=0`, `c=0`
- Pfad: `B1, B2, A2, B3, B1, B2, A3, B3, A4`
  für `a=1`, `b=0`, `c=1`

(c) (1 Punkt) Geben Sie eine formalisierte Testmenge für eine Anweisungsüberdeckung aus *a* an.

PRG 4.2
Folien V5/6
Skript V5/6

- `A1 & A4`: $(a < b \wedge (a + 4) \leq c)$
- `A2 & A4`: $(a \geq b \wedge a \bmod 2 = 1 \wedge (a + 1) \leq c)$
- `A3 & A4`: $(a \geq b \wedge a \bmod 2 \neq 1 \wedge (a - 10) \leq c)$

Allgemeines Verständnis

(d) (6 Punkte) Finden Sie im folgenden Code alle Fehler (12 Stück). Benennen Sie diese und geben Sie die Zeile an, in der der Fehler auftritt.

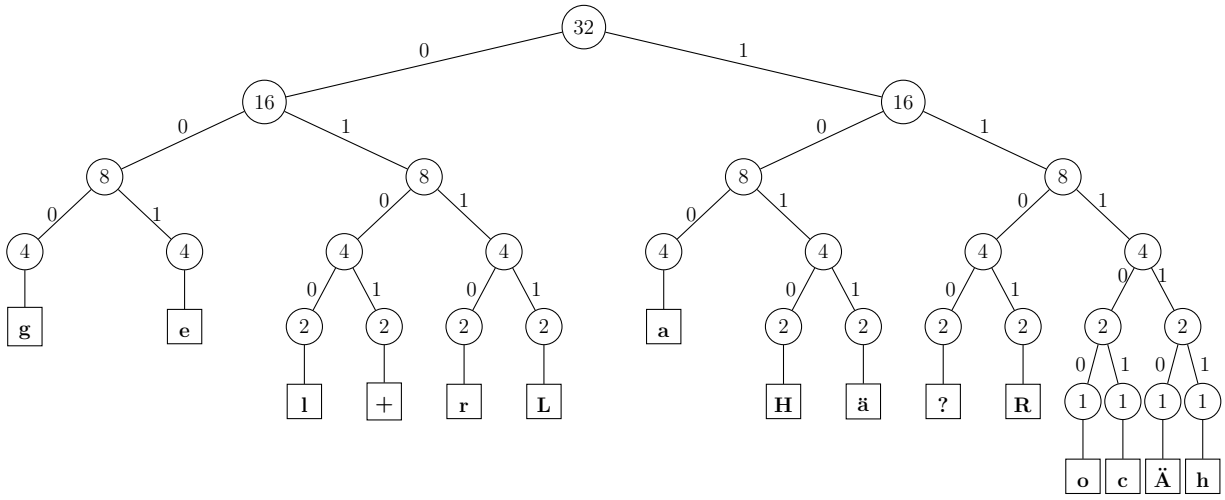
```
1  def merge(left, right):
2      result, i, j = [], 0, 0
3      while i < len(left) and j < len(right):
4          if left[i] <= right[j]:
5              result.append(left[i])
6              i += 1
7          else:
8              result.append(right[j])
9              j += 1
10     return result + left[i:] + right[j:]
11
12 def mergesort(lst):
13     if len(lst) <= 1:
14         return lst
15     middle = len(lst) // 2
16     left = mergesort(lst[:middle])
17     right = mergesort(lst[middle:])
18     return merge(left, right)
```

0.5 Punkte pro gefundenem Fehler.

1. `Zeile 1`: `merge` statt `mergesort`
2. `Zeile 2`: `,` statt `.`
3. `Zeile 3`: `len(left)` statt `left`
4. `Zeile 3`: `and` statt `or`
5. `Zeile 3`: `len(right)` statt `len[right]`
6. `Zeile 6`: `+=` statt `-=`
7. `Zeile 7`: `:` statt `;`
8. `Zeile 13`: `<=` statt `<`
9. `Zeile 14`: `lst` statt `list`
10. `Zeile 15`: `middle` statt `mid`
11. `Zeile 15`: `//` statt `/`
12. `Zeile 17`: `lst[middle:]` statt `lst[middle:0]`

Aufgabe 7: Daten – Information – Wissen **Punkte: 10/ __**

Gegeben sei der folgende Code-Baum einer *Huffman-Kodierung*.



PRG 6.1
EPR 5.3
Folien V9
Skript V9

(a) (2 Punkte) Dekodieren Sie die Nachricht[2].

```
1010111001 1101111110 1001000000 0101100101 1101001000 1000111010
1011110000 0001011011 0100100010 0010010101 0111011111 1011001100
```

```
Hochlager+RegaL+LagerRegalHääÄ??
```

PRG 6.1
EPR 5.3
Folien V9
Skript V9

(b) (1 Punkt) Vervollständigen Sie die Tabelle.

Zeichen z	Anzahl	$H_n(z)$	$I(z)$
a	4	0.125000	3.0
c	1	0.031250	5.0
e	4	0.125000	3.0
ä	2	0.062500	4.0
g	4	0.125000	3.0
H	2	0.062500	4.0
+	2	0.062500	4.0
l	2	0.062500	4.0
o	1	0.031250	5.0
L	2	0.062500	4.0
r	2	0.062500	4.0
h	1	0.031250	5.0
R	2	0.062500	4.0
?	2	0.062500	4.0
Ä	1	0.031250	5.0

0.5 Punkte für die Spalten Anzahl und $H_n(z)$ und 0.5 für $I(z)$

[2] Die in der Nachricht enthaltenen Leerzeichen dienen der Lesbarkeit und sind nicht Bestandteil der Nachricht.

(c) (1 Punkt) Wie definierte Shannon die erwartete Information (*Entropie* H) einer Nachrichtenquelle (*Informationsquelle*) X über einem Zeichenvorrat Ω?

PRG 6.2
EPR 5.3
Folien V9
Skript V9

Lösung:
Eine der Summen genügt.

$$H(X) = E(I(X)) = \sum_{i=1}^{n} p(x_i) \cdot I(x_i) \tag{8}$$

$$= \sum_{i=1}^{n} p(x_i) \cdot log_b(\frac{1}{p(x_i)}) = -\sum_{i=1}^{n} p(x_i) \cdot log_b(p(x_i)) \tag{9}$$

oder

$$-\sum_{i=1}^{|\Omega|} P_i \cdot \log_b(P_i)$$

(d) (1 Punkt) Was lässt sich über den Informationsgehalt $I(x)$ aussagen, wenn $p(x)$ gegen 0 geht? **Begründen Sie ihre Antwort.**

Folien V9
Skript V9

Lösung:
1 Punkt für die Antwort. Der Informationsgehalt geht gegen Unendlich.
1 Punkt für die Begründung. $I(n) = -log_2(p(n)) = log_2(\frac{1}{p(n)})$.
$\Rightarrow \lim_{p(n)\to 0} log_2 \frac{1}{p(n)} = \infty$

(e) (1 Punkt) Warum darf für die Auftrittswahrscheinlichkeit $p(x) = 0$ nicht gelten?

Folien V9
Skript V9

Lösung:
Wenn $p(x) = 0$ gilt ergibt sich für die Berechnung von $I(x)$ eine Division durch Null.
$I(n) = -log_2(p(n)) = log_2(\frac{1}{0}) \Rightarrow \lightning$

(f) (2 Punkte) Wann addiert sich der Informationsgehalt mehrerer Nachrichten? Wann wird er multipliziert?

Folien V9
Skript V9

Lösung:

- Bei unabhängigen Nachrichten addiert sich deren Informationsgehalt.

(g) (2 Punkte) Formulieren Sie eine Grammatik mit Startsymbol `S` als EBNF zur Erzeugung korrekt geklammerter Ausdrücke für die Klammern `()` und `[]`.

PRG 5.5b
Folien V9
Skript V9

Lösung:
`S = {S} | '(', S, ')' | '[', S, ']'.`

Aufgabe 8: Parallele Konstrukte **Punkte: 10/ __**

PRG 9.3b
Folien V14
Skript V14

(a) (2 Punkte) **Was** ist eine *race condition* und **wie** kann sie vermieden werden?

Lösung:

Eine *race condition* kann immer dann auftreten, wenn zwei oder mehrere Prozesse nicht synchronisiert auf die selben Ressourcen zugreifen. Also, der eine Prozesse den anderen überholt. Um *race condition's* zu vermeiden muss der **kritische Abschnitt atomar** sein. D.h. z.B. mit Hilfe von *Semaphoren* oder anderen Mechanismen des gegenseitigen Ausschluss, (**mutual exclusion**) synchronisiert werden.

PRG 9.2a/b
Folien V14
Skript V14

(b) (2 Punkte) Welche Elemente eines Prozesses sind resident und welche sind nicht resident?

Speicherresident:

- Scheduling-Parameter
- Speicherreferenzen: Code-, Daten-, Stackadressen im Speicher
- Signaldaten: Masken, Zustände
- Verschiedenes: Prozesszustand, erwartetes Ereignis, Timerzustand, PID, PID der Eltern, User/Group-IDs

Nicht speicherresident:

- Prozessorzustand: Register, FPU-Register,
- Systemaufruf: Parameter, bisherige Ergebnisse,
- Dateiinfo-Tabelle (file descriptor table)
- Benutzungsinfo: CPU-Zeit, max. Stackgröße,
- Kernel-Stack: Stackplatz für Systemaufrufe des Prozesses

EPR 7.3
Skript V14

(c) (1 Punkt) Nach welcher Formel errechnet sich die maximal erreichbare Beschleunigung (*speedup*) der Programmausführung durch Parallelisierung?

Lösung:

$$speedup = \frac{\text{alte Zeit}}{\text{neue Zeit}} \rightarrow \frac{T_{seq} + T_{par}}{T_{seq} + 0} = 1 + \frac{T_{par}}{T_{seq}}$$

Folien V14
Skript V14

(d) (1 Punkt) Atomare Aktionen lassen sich mittels Hardware implementieren. Welche Verfahren wurden hierzu in der Vorlesung diskutiert?

Lösung:

`fetch_and_add` liest und inkrementierten den Inhalt einer Speicherzelle im selben ununterbrechbaren Speicherzyklus.
`test_and_set`-Instruktion (auch **`test_and_set`** lock genannt).

Skript V15

(e) (1 Punkt) Wodurch wird das Kontrollflussprinzip in imperativen Programmiersprachen implementiert? Nennen Sie Beispiele.

Lösung:

Das Kontrollflussprinzip wird in imperativen Programmiersprachen durch die Kontrollstrukturen implementiert. Z.B `while`, `do`, `for`, `goto`, `if`, `switch` bzw. `case`, ; (Sequenzialisierungsoperator)

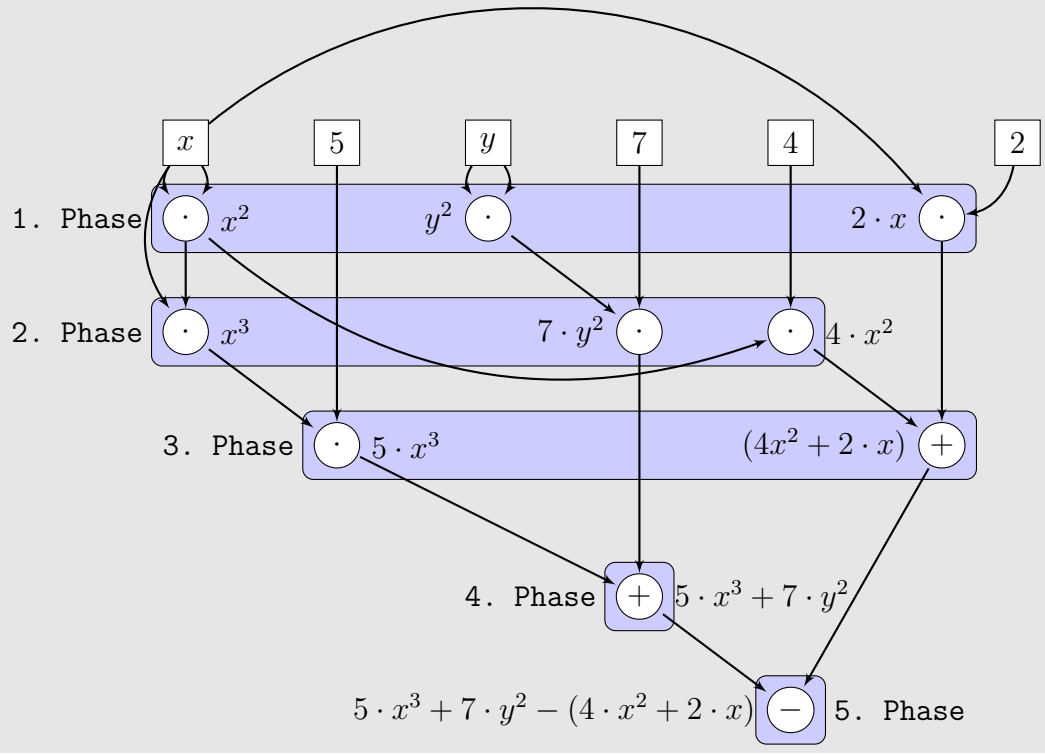
PRG 10.2
Skript V15

(f) (3 Punkte) Zeichnen Sie den Datenflussgraphen für die Funktion

$$5 \cdot x^3 + 7 \cdot y^2 - (4 \cdot x^2 + 2 \cdot x)$$

und geben Sie an, welche der Schritte sich parallel berechnen lassen.

Lösung:

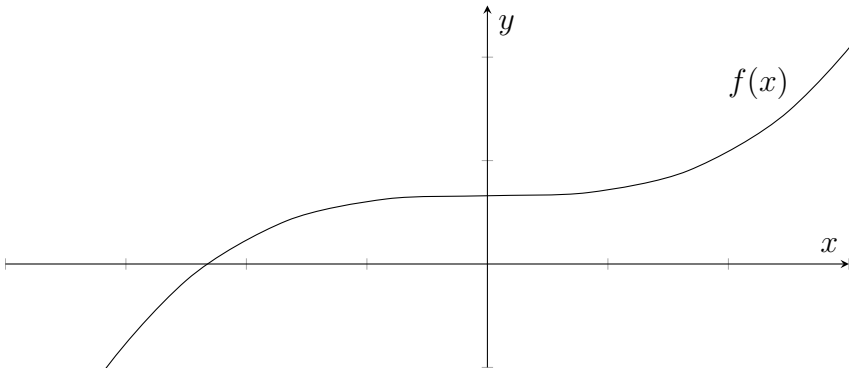


Aufgabe 9: Algorithmenentwurf

Punkte: 10/ __

Folien V15
Skript V15
Transfer

(a) (5 Punkte) Formulieren Sie einen *Divide & Conquer*-Algorithmus als Anweisungsvorschrift (Pseudocode) zur Ermittlung der Nullstelle einer streng monoton Funktion f im Intervall $[a, b]$.



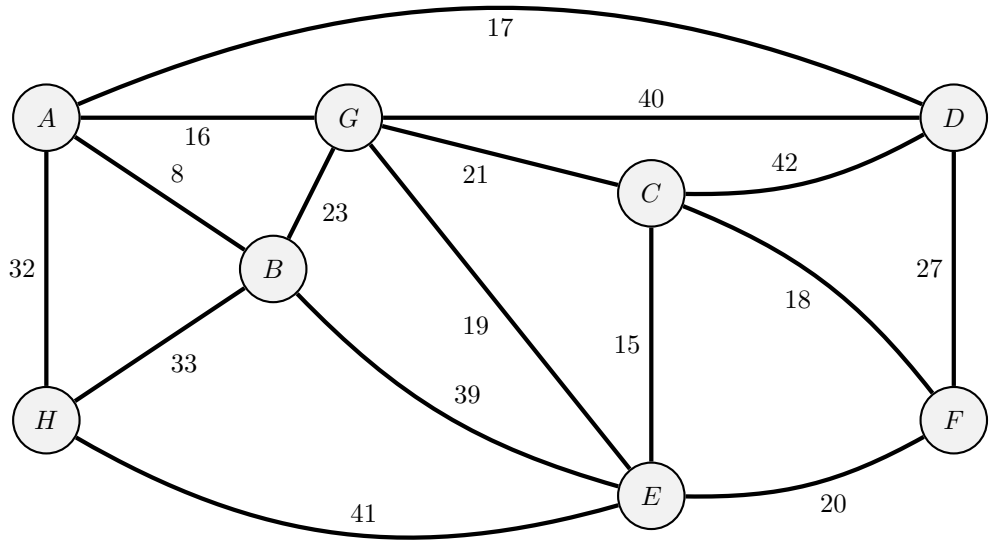
Lösung:

Algorithm 1 Bisektions

```
1: ε ← 0.00...1
2: function BISECTION(f, a, b)
3:     left ← f(a)
4:     right ← f(b)
5:     mid ← (left + right)/2
6:     if |left − right| < ε then
7:         return mid
8:     end if
9:     if left < 0 ∧ mid > 0 then
10:        return BISECTION(f, left, mid)
11:    else
12:        return BISECTION(f, mid, right)
13:    end if
14: end function
```

PRG 10.1
Folien V15
Skript V15

(b) Gegeben sei der folgende Graph $\mathcal{G}$.



1. (2 Punkte) Geben Sie die aufsteigend sortierte Kantenliste an.

Kante	AB	CE	AG	AD	CF	EG	EF	CG
Gewicht	8	15	16	17	18	19	20	21
Kante	BG	DF	AH	BH	BE	DG	EH	CD
Gewicht	23	27	32	33	39	40	41	42

2. (3 Punkte) Ermitteln Sie den minimalen Spannbaum mit dem Algorithmus von *Kruskal*. Geben Sie für jeden Rechenschritt die aktuell besuchte Kante, sowie die Knoten der bereits ermittelten Teilgraphen an.

Kante	Knoten der Teilgraphen
	$\{A\},\{B\},\{C\},\{D\},\{E\},\{F\},\{G\},\{H\}$
(A,B)	$\{A,B\},\{C\},\{D\},\{E\},\{F\},\{G\},\{H\}$
(C,E)	$\{A,B\},\{C,E\},\{D\},\{F\},\{G\},\{H\}$
(A,G)	$\{A,B,G\},\{C,E\},\{D\},\{F\},\{H\}$
(A,D)	$\{A,B,D,G\},\{C,E\},\{F\},\{H\}$
(C,F)	$\{A,B,D,G\},\{C,E,F\},\{H\}$
(E,G)	$\{A,B,C,D,E,F,G\},\{H\}$
(A,H)	$\{A,B,C,D,E,F,G,H\}$

Falls Sie auf der Titelseite das Kreuz für 9 CP's gesetzt haben entfällt diese Aufgabe für Sie.

Aufgabe 10: Softwareentwurf **Punkte: 10/ __**

PRG 11.2
Folien V16
Skript V16

(a) (6 Punkte) Nennen und erklären Sie die einzelnen Phasen des Wasserfallmodels.

Lösung:

Je Phase 0.5 Punkte für den Name und 0.5 für die Beschreibung.

1. **Phase Durchführbarkeitsanalyse**
   Der Auftragnehmer führt eine Durchführbarkeitsanalyse durch, in der er das Problem noch klar formuliert, Kriterien zur Lösungsfindung definiert, mögliche Skizzen und Empfehlungen zu deren Durchführung angibt.

2. **Phase Problemformulierung**
   Die Problemformulierung untergliedert sich in: **Anforderungsanalyse** (präzise Spezifikation der **Anforderungsspezifikation**)
   Dem **Projektplan**:detaillierter Plan zur Entwicklung des Produktes (Budget, Zeitplan, Methodik, Training)

3. **Phase Design**
   Im **Designschritt** wird zum einen die Gesamtstruktur des Systems spezifiziert, sowie die Arbeitsweise der Einzelkomponenten festgelegt.

4. **Phase Implementierung**
   Im Implementierungsschritt werden die einzelnen Komponenten konstruiert, wobei sichergestellt sein muss, dass diese unabhängig vom Gesamtsystem arbeiten. Dann werden diese zum Gesamtsystem zusammengesetzt.

5. **Phase Testen**
   Testen ob das Programm den Spezifikationen entspricht und korrekt läuft.

6. **Phase Abgabe**
   Nach Fertigstellung des Produktes erfolgt die **Auslieferung** an den Kunden.

(b) (1 Punkt) Worin besteht der Sinn der Modularisierung von Softwaresystemen? Was bildet die Basis für diese Modularisierung?

Folien V16
Skript V16

Lösung:

Je 0.5 Punkte

Der Sinn der Modularisierung besteht in der **Zusammenfassung von Typen, Daten und Algorithmen** in logische Einheiten. Die Grundlage hierfür bilden **Schnittstellen.**

(c) (2 Punkte) Mit welchen Kriterien lässt sich die Qualität eines modularen Entwurfs *„messen“*? Was bezeichnen diese?

Folien V16
Skript V16

Lösung:

Je 0.5 Punkte für den Begriff und die Erklärung

Durch Kohäsion und Kopplung der Module.
Kohäsion bezeichnet die Bindung der Komponenten innerhalb eines Moduls.
Kopplung bezeichnet die Verbundenheit zwischen den Modulen

(d) (1 Punkt) Wodurch wird die Adaptierbarkeit eines Moduls begünstigt?

Folien V16
Skript V16

Lösung:

Je 0.5 Punkte

Die Adaptierbarkeit eine Moduls wird durch eine *„klare“* **Strukturierung**, sowie eine auf die Implementierung abgestimmte und eine **qualitativ anspruchsvolle Dokumentation** begünstigt.