1 Wissensfragen (6 x 0.5 = 3 Punkte)

Welche der folgenden Aussagen ist richtig, welche falsch? (Bitte zutreffendes ankreuzen)

a) $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ richtig ◯ falsch ◯

b) $\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$ richtig ◯ falsch ◯

c) $\prod_{i=0}^{n-1} i = (n-1)!$ richtig ◯ falsch ◯

d) Eine Prozedur die sich mindestens einmal selbst aufruft, ist rekursiv. richtig ◯ falsch ◯

e) In Ada sind Lebensdauer und Sichtbarkeitsbereich einer Variable immer gleich. richtig ◯ falsch ◯

f) In Ada darf keine Berechnung außerhalb der *range* eines Untertyps durchgeführt werden. richtig ◯ falsch ◯

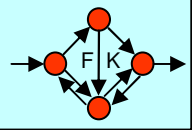
2 BNF (3 Punkte)

Gegeben ist folgender Auszug aus der Syntax einer Programmiersprache:

```
<Start> ::= <if-Anweisung> | <case_Anweisung> | <Schleife> | <Block>
<if-Anweisung> ::= if <Bedingung> then <Anweisungsfolge> end if; |
                   if <Bedingung> then <Anweisungsfolge>
                     else <Anweisungsfolge> end if;
<Bedingung> ::= x>11
<Anweisungsfolge> ::= <Anweisung> | <Anweisung> <Anweisungsfolge>
<Anweisung> ::= x:=3; | x:=11; | x:=0;
```

Leiten Sie folgende Anweisung mit Hilfe der obigen Regeln ab und stellen Sie die Ableitung in Form eines Ableitungsbaums dar.

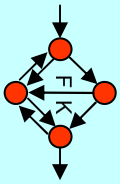
```
if x>11 then x:=3; else x:=0; end if;
```



3 Was wird ausgegeben (9 Punkte)

Analysieren Sie das folgende Ada-Programm. Welche Ausgabe wird erzeugt?

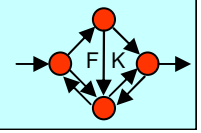
```
with Ada.Text_IO, Ada.Integer_Text_IO;
use Ada.Text_IO, Ada.Integer_Text_IO;

procedure Aufg3 is
a, b, c: Integer;

procedure Write3Int(x, y, z: Integer) is
begin
Put(x,3);
Put(y,3);
Put(z,3);
New_Line;
end Write3Int;

procedure P1(b: in out Integer; d,c: Integer) is
begin
a:= (c + a) * d;
b:= a + c - d;
Write3Int(a,b,c);
end P1;

begin
a:=1; b:=2; c:=3;
Write3Int(a,b,c);
P1(a, b, c);
Write3Int(a,b,c);
```



4 Fehler finden (4 x 1 + 2 = 6 Punkte)

Finden Sie im folgenden Codefragment mindestens vier Fehler und kommentieren Sie diese in einem kurzen Satz. Schreiben Sie das Programm so um, dass es (eingebettet in ein Hauptprogramm) compilierbar wird.

```
subtype sb is Integer range 5..1000;  -- Zeile 1
max1  : constant integer:=2;          -- Zeile 2
max2 : constant integer:=3            -- Zeile 3
feld : array (1..max2,1..max1)        -- Zeile 4
   of sd : ( (100,100,101),           -- Zeile 5
             (10,-1,-1,10) );         -- Zeile 6
begin  put(feld(2,3)); end;           -- Zeile 7
```



5 Rekursion (3 + 2 + 2 = 7 Punkte)

Das Ada-Programm `Aufg5` beinhaltet die rekursive Prozedur `Eoro(i:natural)`.

- Was berechnet die rekursive Prozedur?
- Was wird am Bildschirm ausgegeben?
- Wie oft wird die rekursive Prozedur mit dem Startwert 33 aufgerufen?

```
with Ada.Text_IO, Ada.Integer_Text_IO;
use Ada.Text_IO, Ada.Integer_Text_IO;

procedure Aufg5 is

procedure Eoro(i : natural) is
begin
  if i=0 then
    Put_line("this number is e");
  elsif i=1 then
    Put_line("this number is o");
  else
    Eoro(i-2);
  end if;
end Eoro;

begin
  Eoro(33);
  Eoro(170);
  Eoro(-12);
end Aufg5;
```

6 Datenstruktur definieren (3 Punkte)

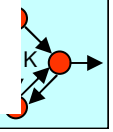
Erstellen Sie eine Datenstruktur für Zugwaggons. Für jeden Zugwaggon sollen die maximal zulässige Ladung in Tonnen (ganzzahlig), der Herstellername (maximale Länge ist 20) und das Jahr des letzten technischen Checks gespeichert werden. Definieren Sie die Datenstruktur so, dass keine falschen Zuweisungen möglich sind.

7 Größenordnung (+1 oder -1 Punkt)

Was ist $0.05 * 0.02$?

Falls Sie das falsche ankreuzen, wird ein Punkt abgezogen!

0.01	◯
0.02	◯
0.001	◯
0.0001	◯
0.002	◯
0.1	◯



Aufgabe 1 (Sortieren)

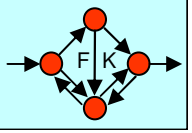
Wir wollen ein Zahlenfeld der Länge n sortieren. Zu diesem Zweck wird das Minimum gesucht. Das Minimum wird dann mit dem ersten Element des Feldes vertauscht. Somit erhält man links ein sortiertes Teilfeld der Länge 1 und rechts ein unsortiertes der Länge *n-1*. Anschließend wird der Algorithmus auf das unsortierte Teilfeld angewendet. Dieses Verfahren heißt Selectionsort. Formulieren Sie ein Computerprogramm, welcher ein Zahlenfeld Feld mit *n* Elementen mit dem Sortierverfahren Selectionsort sortiert. Wie viele Vergleiche benötigt man dabei abhängig von der Anzahl *n*?

Aufgabe 3 (Algorithmusanalyse)

Seien *a* und *b* positive ganze Zahlen und *Q* eine wie folgt definierte Funktion:

$$Q(a,b) = \begin{cases} 0 & \text{für } a < b \\ Q(a-b,b) + 1 & \text{für } a >= b \end{cases}$$

- Formulieren Sie ein rekursives Unterprogramm zur Berechnung von *Q*.
- Was berechnet diese Funktion?



Aufgabe 2 (Rekursion Binomialkoeffizienten)

Vom Pascalschen Dreieck seien folgende Eigenschaften bekannt:

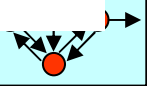
$$\binom{n}{0} = \binom{n}{n} = 1 \qquad \text{für } n \geq 0 \qquad \text{(Startwerte)}$$

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \qquad \text{für } n > 0,\ 0 < k < n \qquad \underline{\text{Rekursion}}$$

Schreiben Sie ein rekursives Programm in einer beliebigen Programmiersprache auf, welches $\binom{n}{k}$ berechnet.

Aufgabe 4 (Multiplikation)

Schreiben Sie in einer beliebigen Programmiersprache eine rekursive Funktion, die das Produkt zweier Zahlen *a, b* vom Typ natural berechnet, ohne jedoch die Multiplikation zu verwenden. Verwenden Sie dabei die Beziehung $a \cdot b = a \cdot (b - 1) + a$.



Aufgabe 5 (Fibonaccizahlen)

Leonardo Fibonacci hat 1202 in seinen Buch "Liber abaci" die Frage gestellt, wie viele Kaninchen-Paare es nach n Jahren gebe, wenn man im Jahr 1 mit einem Paar beginnt, jedes Paar vom zweiten Jahr an ein weiteres Paar Nachwuchs hat und die Kaninchen eine unendliche Lebensdauer haben. Offenbar entsteht dabei die Zahlenfolge:

Jahr	1	2	3	4	5	6	7	8	...
Anzahl der Paare	1	1	2	3	5	8	13	21	...

Diese *Fibonacci-Zahlen* lassen sich wie folgt definieren:

$$fib(n) = \begin{cases} 1 & \text{für } n = 1 \text{ bzw. } n = 2 \\ fib(n\text{-}1) + fib(n\text{-}2) & \text{für } n > 2 \end{cases}$$

Programmieren Sie die zugehörige rekursive Funktion auf.

Aufgabe 6 (ehemalige Prüfungsaufgabe)

Schreiben Sie eine rekursive Funktionsprozedur zur Berechnung der n-ten Potenz einer reellen Zahl x mit x vom Typ FLOAT und n vom Typ INTEGER. Um die Zahl der Multiplikationen gering zu halten, sind folgende Zusammenhänge zu verwenden:

$$x^n = \frac{1}{x^{-n}} \text{ für } n < 0$$

$$x^n = x^{n/2} \cdot x^{n/2} \text{ für gerade } n$$

$$x^n = x^{(n-1)/2} \cdot x^{(n-1)/2} \cdot x \text{ für ungerade } n$$

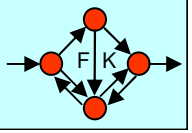
Aufgabe 7: (Der Algorithmus von Euklid)

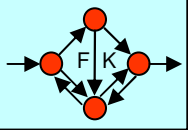
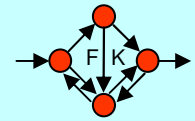
Der Euklidische Algorithmus berechnet den größten gemeinsamen Teiler von zwei natürlichen Zahlen. Der Algorithmus wurde bereits ca. 300 v.Chr. von Euklid beschrieben. Wir werden im Folgenden eine rekursive Variante des Algorithmus angeben.
Seien $a \in \mathbb{N}_0$, $b \in \mathbb{N}$. Dann gilt $ggT(a, b) = ggT(b, a \bmod b)$ und ggT(a,0)=a.

Schreiben Sie ein rekursives Programm in einer beliebigen Programmiersprache auf, welches $ggT(a, b)$ berechnet.

Aufgabe 8: (rekursive Decent Parser)

Schreiben Sie einen Algorithmus, der rekursiv Worte der Form $a^n b^n$ erkennt. Erklären Sie den Begriff Linksrekursion. Schreiben Sie einen Algorithmus, der arithmetische Ausdrücke auswertet. Dabei soll ‚Punkt vor Strich' zunächst außer Acht gelassen werden.




F
K

Kapitel 2: Inhaltsübersicht