

Aufgabensammlung

PVK Informatik

GianAndrea Müller - 2018

basierend auf Prüfungen vom D-INFK

1 Typen und Werte / Types and values (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) $8 + 5 / 3$

1.5 P

Typ/Type

Wert/Value

(b) $1.0 * 0.1 == 0.1 * 1.0$

1.5 P

Typ/Type

Wert/Value

(c) $1e1 * 2e2$

1.5 P

Typ/Type

Wert/Value

(d) $5 / 2.0f + 5 / 2.0$

1.5 P

Typ/Type

Wert/Value

(e) $!false || true \&\& false$

1.5 P

Typ/Type

Wert/Value

(f) $17 \% 5u$

1.5 P

Typ/Type

Wert/Value

1 Typen und Werte / Types and values (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) $8 + 5 / 3$

1.5 P

Typ/Type

int

Wert/Value

9

(b) $1.0 * 0.1 == 0.1 * 1.0$

1.5 P

Typ/Type

bool

Wert/Value

true

(c) $1e1 * 2e2$

1.5 P

Typ/Type

double

Wert/Value

2000

(d) $5 / 2.0f + 5 / 2.0$

1.5 P

Typ/Type

double

Wert/Value

5

(e) $!false || true \&\& false$

1.5 P

Typ/Type

bool

Wert/Value

true

(f) $17 \% 5u$

1.5 P

Typ/Type

unsigned int

Wert/Value

2

1 Typen und Werte (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) $(1 \ || \ 1) \ \&\& \ !(1 \ \&\& \ 1)$

1.5 P

Typ/Type

Wert/Value

(b) $1.0\text{e}2\text{f} \ / \ 2.0\text{e}3\text{f}$

1.5 P

Typ/Type

Wert/Value

(c) $0\text{x}2 \ * \ 0\text{x}\text{f}$

1.5 P

Typ/Type

Wert/Value

(d) $1.0\text{f} \ == \ 1$

1.5 P

Typ/Type

Wert/Value

(e) $10 \ \% \ 4 \ * \ 5$

1.5 P

Typ/Type

Wert/Value

(f) $15 \ / \ 2 \ / \ 5 \ * \ 2$

1.5 P

Typ/Type

Wert/Value

1 Typen und Werte (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) `(1 || 1) && !(1 && 1)`

1.5 P

Typ/Type

bool

Wert/Value

false

(b) `1.0e2f / 2.0e3f`

1.5 P

Typ/Type

float

Wert/Value

0.05

(c) `0x2 * 0xf`

1.5 P

Typ/Type

int

Wert/Value

30

(d) `1.0f == 1`

1.5 P

Typ/Type

bool

Wert/Value

true

(e) `10 % 4 * 5`

1.5 P

Typ/Type

int

Wert/Value

10

(f) `15 / 2 / 5 * 2`

1.5 P

Typ/Type

int

Wert/Value

2

1 Typen und Werte (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) $5 / 2 * 2.0$

1.5 P

Typ/*Type*

Wert/*Value*

(b) $0xa + 0xb$

1.5 P

Typ/*Type*

Wert/*Value*

(c) $0 / 1 == 0 \ || \ 1 / 0 == 1$

1.5 P

Typ/*Type*

Wert/*Value*

(d) $1.0e4f / 1.0e2f$

1.5 P

Typ/*Type*

Wert/*Value*

(e) $3 + 17 \% 4$

1.5 P

Typ/*Type*

Wert/*Value*

(f) $0.1f == 0.1$

1.5 P

Typ/*Type*

Wert/*Value*

1 Typen und Werte (9 Punkte)

Geben Sie für jeden der sechs Ausdrücke unten jeweils C++-Typ (0.5 P) und Wert (1 P) an! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! *For each of the six expressions below, provide the C++ type (0.5 P) and value (1 P)! For floating point numbers, assume the IEEE 754 standard!*

(a) $5 / 2 * 2.0$

1.5 P

Typ/Type

double

Wert/Value

4

(b) $0xa + 0xb$

1.5 P

Typ/Type

int

Wert/Value

21

(c) $0 / 1 == 0 \ || \ 1 / 0 == 1$

1.5 P

Typ/Type

bool

Wert/Value

true

(d) $1.0e4f / 1.0e2f$

1.5 P

Typ/Type

float

Wert/Value

100

(e) $3 + 17 \% 4$

1.5 P

Typ/Type

int

Wert/Value

4

(f) $0.1f == 0.1$

1.5 P

Typ/Type

bool

Wert/Value

false

8 Argumenttypen (8 Punkte)

Auf der rechten Seite finden Sie Funktionsdeklarationen, bei denen jeweils die Argumenttypen weggelassen sind. In jedem Fall spezifiziert die Vorbedingung das Verhalten der Funktion. Ihre Aufgabe ist es, die dazu passenden Argumenttypen einzusetzen. Beantworten Sie die Fragen auf der rechten Seite!

On the right-hand side, you find function declarations in which the argument types have been omitted. In each case, the precondition specifies the behavior of the function. Your task is to provide the matching argument types. Answer the questions on the right-hand side!

-
- (a) Ergänzen Sie korrekte Argumenttypen! Provide matching argument types!

2 P

```
// POST: erhoeht den Wert der Variablen balance um amount und gibt
//       den neuen Wert von balance zurueck
//       increases the value of the variable balance by amount and
//       returns the new value of balance
float top_up (  balance,  amount);
```

-
- (b) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// PRE: [begin, end) ist ein nichtleerer Bereich von double-Werten
//       [begin, end) is a valid nonempty range of double-values
// POST: gibt den Durchschnitt der Werte in [begin, end) zurueck
//       returns the average of all values in [begin, end)
double average (  begin,  end);
```

-
- (c) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// POST: der Bereich [begin, end) von ganzen Zahlen wird in
//       aufsteigender Reihenfolge sortiert
//       the range [begin end) of integers is sorted in
//       increasing order
void sort (  begin,  end);
```

-
- (d) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// POST: gibt zurueck, ob die natuerliche Zahl n eine Primzahl
//       ist; falls nein, wird factor auf einen echten Teiler
//       von n gesetzt
//       returns whether the natural number n is prime. If not,
//       factor is set to a proper divisor of n
bool is_prime (  n,  factor);
```

- (a) Ergänzen Sie korrekte Argumenttypen! Provide matching argument types!

2 P

```
// POST: erhoeht den Wert der Variablen balance um amount und gibt
//       den neuen Wert von balance zurueck
//       increases the value of the variable balance by amount and
//       returns the new value of balance
float top_up ( float& balance, float amount );
```

- (b) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// PRE: [begin, end) ist ein nichtleerer Bereich von double-Werten
//       [begin, end) is a valid nonempty range of double-values
// POST: gibt den Durchschnitt der Werte in [begin, end) zurueck
//       returns the average of all values in [begin, end)
double average ( const double* begin, const double* end );
```

- (c) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// POST: der Bereich [begin, end) von ganzen Zahlen wird in
//       aufsteigender Reihenfolge sortiert
//       the range [begin end) of integers is sorted in
//       increasing order
void sort ( int* begin, int* end );
```

- (d) Ergänzen Sie die korrekten Argumenttypen! Provide the matching argument types!

2 P

```
// POST: gibt zurueck, ob die natuerliche Zahl n eine Primzahl
//       ist; falls nein, wird factor auf einen echten Teiler
//       von n gesetzt
//       returns whether the natural number n is prime. If not,
//       factor is set to a proper divisor of n
bool is_prime ( unsigned int n, unsigned int& factor );
```

Aufgabe 7: Referenztypen und Pointer (7P)

Das Programm auf der rechten Seite definiert eine Datenstruktur für Datumsangaben und stellt weitere Operationen auf dieser Struktur bereit. Beantworten Sie für dieses Programm untenstehende Fragen.

The program on the right hand side defines a data structure to store dates and provides additional operations that operate on this structure. Answer the questions below with regard to this program.

/4P

- (a) In der untenstehenden Tabelle sehen Sie zwei verschiedene Ausgaben des Programms, sowie Spalten mit Markierungen, die diejenigen in der Klasse `DateList` entsprechen. Füllen Sie die Tabelle mit Typdeklarationen oder Ausdrücken, so dass das Programm die entsprechende Ausgabe erzeugt (pro Zeile).

In the table below, you see two different outputs created by the program and columns with markers that point to locations within the class `DateList`. Fill in the markers with type declarations or expressions, such that the program creates the output as specified (per row).

Ausgabe: / <i>Output:</i>	A	B	C	D
1.4.2001 31.7.2013				
23.8.2017 31.7.2013				

/2P

- (b) Betrachten Sie die Funktion `createDate`, welche eine Datumstruktur deklariert, initialisiert und zurückgibt. Beantworten Sie folgende Frage (je ein Satz): (i) Welchen Fehler hat diese Funktion und (ii) wie könnte man die Funktion korrigieren?

Look at the function `createDate` that declares and initializes a date structure and returns it. Answer the following questions (each one sentence): (i) What error does this function have, and (ii) how could the function be fixed ?

(i)

(ii)

/1P

- (c) Betrachten Sie die Funktionen `f1`, `f2`, `f3`, welche jeweils eine Datumsstruktur als Parameter übernehmen. Der Parameter ist ein Referenztyp oder ein Pointer mit unterschiedlicher `const`-Eigenschaft. Markieren Sie alle Funktionen die korrekt kompilieren.

Look at the function `f1`, `f2`, `f3` that receive a date structure as parameter. The parameters are either reference types or pointers and have different `const` properties. Mark those functions that compile correctly.

☐ `f1` ☐ `f2` ☐ `f3`

```
#include <iostream>
#include <vector>

struct Date {
    int day; int month; int year;

    void set(int d, int m, int y) { day = d; month = m; year = y; }
    void print() {
        std::cout << day << "." << month << "." << year << "\n";
    }
};

class DateList {
    std::vector<A> dates;
public:
    void add(B date) { dates.push_back(C); }
    void print() {
        for(std::vector<A>::iterator it=dates.begin(); it!=dates.end(); ++it) {
            D.print();
        }
    }
};

int main() {
    DateList list; Date d1; Date d2;

    d1.set(1, 4, 2001);
    list.add(d1);
    d2.set(31,7, 2013);
    list.add(d2);
    d1.set(23, 8, 2017);

    list.print();
    return 0;
}

Date* createDate(int day, int month, int year) {
    Date d;
    d.set(day, month, year);
    return &d;
}

void f1(const Date& date) { date.year = 0; }

void f2(const Date* date) { date->year = 0; }

void f3(Date* const date) { date->year = 0; }
```

Aufgabe 7: Referenztypen und Pointer (7P)

Das Programm auf der rechten Seite definiert eine Datenstruktur für Datumsangaben und stellt weitere Operationen auf dieser Struktur bereit. Beantworten Sie für dieses Programm untenstehende Fragen.

The program on the right hand side defines a data structure to store dates and provides additional operations that operate on this structure. Answer the questions below with regard to this program.

/4P

- (a) In der untenstehenden Tabelle sehen Sie zwei verschiedene Ausgaben des Programms, sowie Spalten mit Markierungen, die diejenigen in der Klasse `DateList` entsprechen. Füllen Sie die Tabelle mit Typdeklarationen oder Ausdrücken, so dass das Programm die entsprechende Ausgabe erzeugt (pro Zeile).

In the table below, you see two different outputs created by the program and columns with markers that point to locations within the class `DateList`. Fill in the markers with type declarations or expressions, such that the program creates the output as specified (per row).

Ausgabe: / <i>Output:</i>	A	B	C	D
1.4.2001 31.7.2013	Date	Date& or Date	date	*it
23.8.2017 31.7.2013	Date*	Date&	&date	**it

/2P

- (b) Betrachten Sie die Funktion `createDate`, welche eine Datumstruktur deklariert, initialisiert und zurückgibt. Beantworten Sie folgende Frage (je ein Satz): (i) Welchen Fehler hat diese Funktion und (ii) wie könnte man die Funktion korrigieren?

Look at the function `createDate` that declares and initializes a date structure and returns it. Answer the following questions (each one sentence): (i) What error does this function have, and (ii) how could the function be fixed ?

- (i) Returns pointer to deallocated object instance.

- (ii) `Date* d = new(day, month, year);`

/1P

- (c) Betrachten Sie die Funktionen `f1`, `f2`, `f3`, welche jeweils eine Datumstruktur als Parameter übernehmen. Der Parameter ist ein Referenztyp oder ein Pointer mit unterschiedlicher `const`-Eigenschaft. Markieren Sie alle Funktionen die korrekt kompilieren.

Look at the function `f1`, `f2`, `f3` that receive a date structure as parameter. The parameters are either reference types or pointers and have different `const` properties. Mark those functions that compile correctly.

☐ f1 ☐ f2 ☒ f3

2 Typen und Werte II (6 Punkte)

Geben Sie für jeden der drei Ausdrücke auf der rechten Seite jeweils C++-Typ und Wert an!

Wir merken an, dass z.B. `integer` oder `boolean` keine C++-Typen sind und als falsche Antworten gewertet werden.

Das Array `a` sei deklariert und initialisiert wie folgt.

```
int a[] = {1, 3, 4};
```

For each of the 3 expressions on the right, provide the C++ type and value!

Note that, for example, `integer` or `boolean` are not C++ types and will be considered as incorrect answers.

Array `a` has been declared and initialized as shown above.

`&a[0] + 1 == &a[1]`

1 P

Typ/Type:

`&a[0] + 1 == &a[1]`

1 P

Wert/Value:

`a[a[2] % a[1]]`

1 P

Typ/Type:

`a[a[2] % a[1]]`

1 P

Wert/Value:

`*(a + 2) / *a * 3`

1 P

Typ/Type:

`*(a + 2) / *a * 3`

1 P

Wert/Value:

```
&a[0] + 1 == &a[1]
```

1 P

Typ/Type:

```
&a[0] + 1 == &a[1]
```

1 P

Wert/Value:

```
a[a[2] % a[1]]
```

1 P

Typ/Type:

```
a[a[2] % a[1]]
```

1 P

Wert/Value:

```
*(a + 2) / *a * 3
```

1 P

Typ/Type:

```
*(a + 2) / *a * 3
```

1 P

Wert/Value:

2 Typen und Werte (Structs und Pointers) (7.5 Punkte)

Geben Sie für jeden der Ausdrücke auf der rechten Seite jeweils den C++-Typ (0.5 P) und Wert (1 P) an!

Die verwendeten Variablen haben Typ und Wert wie am Ende der Funktion `f`.

```
struct P {
    int a;
    unsigned int w;
    P* s;

    P(int A, unsigned int W, P* S): a(A), w(W), s(S)
    {}

    P(): a(4), w(13), s(0)
    {}
};

void f(){
    P p;
    P* ps = new P(6, 20, &p);
    P* pt = new P(3, 17, ps);
    P& q = p;
    q.s = pt;

    int a = 10;
    int b = 2;
    int &c = a;
    ++c;

    int A[3] = {1,0,2};
    int *X = &A[0];

    // type and value of the variables at this point
}
```

For each of the expressions on the right, provide its C++ type (0.5 P) and value (1 P)!

Assume that the used variables have type and value as at the end of the function `f` above.

(a) a 1.5 P

Typ/TypeWert/*Value*

(b) x[A[1]] 1.5 P

Typ/TypeWert/*Value*

(c) p.w 1.5 P

Typ/TypeWert/*Value*

(d) (*ps).a 1.5 P

Typ/TypeWert/*Value*

(e) p.s->a 1.5 P

Typ/TypeWert/*Value*

-
- (a) a 1.5 P

Typ/Type	Wert/Value
int	11

-
- (b) x[A[1]] 1.5 P

Typ/Type	Wert/Value
int	1

-
- (c) p.w 1.5 P

Typ/Type	Wert/Value
unsigned int	13

-
- (d) (*ps).a 1.5 P

Typ/Type	Wert/Value
int	6

-
- (e) p.s->a 1.5 P

Typ/Type	Wert/Value
int	3

1/2 Punkt für jeden korrekten Typ, 1P für jeden korrekten Wert

5 Fehlersuche (6 Punkte)

Betrachten Sie das folgende fehlerhafte Programm mit Deklaration und Benutzung der Klasse Clock. Die Datenmember und Definitionen der Memberfunktionen sind für diese Aufgabe irrelevant und deshalb nicht angegeben. Beantworten Sie die Fragen auf der rechten Seite! Jede Zeilennummer darf nur einmal angegeben werden, auch wenn die Zeile mehrere (gleichartige) Fehler enthält.

```
1  class Clock {
2
3      // PRE: h < 24, m < 60, s < 60
4      // POST: creates a clock showing time h:m:s
5      Clock (unsigned int h, unsigned int m, unsigned int s);
6
7      // POST: advances the clock's time by one second
8      void tick () const;
9
10     // POST: h, m, s are set to the clock's hour, minute and second
11     void time (unsigned int h, unsigned int m, unsigned int s) const;
12
13     private:
14         // data members to store the time
15 };
16
17 main() {
18     Clock c = (22, 59, 59);
19     c.tick();
20
21     unsigned int h; unsigned int m; unsigned int s;
22     time (h, m, s); // 23:00:00
23
24     return 0;
25 }
```

The above program with declaration and usage of the class Clock contain some errors. The data members and definitions of the member functions are irrelevant for this assignment and are therefore omitted. Answer the questions on the right-hand side! Each line number can be specified only once, even if the line contains several errors (of the same type).

-
- (a) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (b) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (c) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (d) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (e) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (f) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A:

-
- (a) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 2: the public keyword is missing

-
- (b) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 8: tick is not a const member function, since it changes the time

-
- (c) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 11: the arguments must be of reference type

-
- (d) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 17: the main function must have return type int

-
- (e) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 18: illegaler Konstruktoraufruf

-
- (f) Geben Sie die Nummer einer fehlerhaften Zeile an und beschreiben Sie kurz den Fehler! 1 P
Specify the number of a line containing an error, and briefly describe the error!

A: 22: the function time must be called for the variable c

4 Zeichenketten umdrehen (6 Punkte)

Das folgende Programm definiert und benutzt eine Funktion zur Ausgabe einer Zeichenkette in umgekehrter Reihenfolge. Beantworten Sie die Fragen auf der rechten Seite so, dass sich eine korrekte Implementierung ergibt.

```
// PRE: [begin, end) is a valid range of characters
// POST: [begin, end) is written to std::cout in reversed order
void print_reverse (const char* begin, const char* end)
{
    for (const char* i = A ; B ; C )
        std::cout << D ;
}%>>

int main()
{
    char language[] = {'n','o',' ',' ','C','+', '+'};
    print_reverse ( E , F ); // ++C on
    std::cout << '\n';
    return 0;
}
```

The program above defines and uses a function for outputting a sequence of characters in reversed order. Read and answer the questions on the right side such that you get a correct implementation!

Welcher Ausdruck muss bei A eingesetzt werden?

1 P

Fill in the expression for A.

Welcher Ausdruck muss bei B eingesetzt werden?

1 P

Fill in the expression for B.

Welcher Ausdruck muss bei C eingesetzt werden?

1 P

Fill in the expression for C.

Welcher Ausdruck muss bei D eingesetzt werden?

1 P

Fill in the expression for D.

Welcher Ausdruck muss bei E eingesetzt werden?

1 P

Fill in the expression for E.

Welcher Ausdruck muss bei F eingesetzt werden?

1 P

Fill in the expression for F.

Welcher Ausdruck muss bei A eingesetzt werden?

1 P

Fill in the expression for A.

```
end, alternatively: end-1
```

Welcher Ausdruck muss bei B eingesetzt werden?

1 P

Fill in the expression for B.

```
i > begin, alternatively: i >= begin
```

Welcher Ausdruck muss bei C eingesetzt werden?

1 P

Fill in the expression for C.

```
--i
```

Welcher Ausdruck muss bei D eingesetzt werden?

1 P

Fill in the expression for D.

```
*(i-1), alternatively: *i
```

Welcher Ausdruck muss bei E eingesetzt werden?

1 P

Fill in the expression for E.

```
language
```

Welcher Ausdruck muss bei F eingesetzt werden?

1 P

Fill in the expression for F.

```
language+6
```

2 Schleifenausgaben / Loop outputs (6 Punkte)

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe an, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"!

For each of the following three code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!

- (a)

```
for (double d = 0.25; d != 16; d = 2*(d+1))  
    std::cout << d << ", ";
```

2 P

Ausgabe/Output:

- (b)

```
int a[] = {1, 2, -2, -1};  
int* p = a;  
do {  
    std::cout << *p << ", ";  
    p += *p;  
} while (p != a);
```

Ausgabe/Output:

2 P

- (c)

```
unsigned int n = 12;  
while (n > 1) {  
    std::cout << n << ", ";  
    if (n % 2 == 0)  
        n = n / 2;  
    else  
        n = 2 * n;  
}
```

Ausgabe/Output:

2 P

2 Schleifenausgaben / Loop outputs (6 Punkte)

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe an, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"!

For each of the following three code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!

- (a)

```
for (double d = 0.25; d != 16; d = 2*(d+1))
    std::cout << d << ", ";
```

2 P

Ausgabe/Output:

0.25, 2.5, 7,

- (b)

```
int a[] = {1, 2, -2, -1};
int* p = a;
do {
    std::cout << *p << ", ";
    p += *p;
} while (p != a);
```

Ausgabe/Output:

1, 2, -1, -2,

2 P

- (c)

```
unsigned int n = 12;
while (n > 1) {
    std::cout << n << ", ";
    if (n % 2 == 0)
        n = n / 2;
    else
        n = 2 * n;
}
```

Ausgabe/Output:

infinite loop

2 P

2 Schleifenausgaben (6 Punkte)

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe aus, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"! *For each of the following three code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!*

(a)

```
double d = 2.0;
do {
    std::cout << (--d)++;
} while (d < 2.0);
```

Ausgabe/Output:

2 P

(b)

```
double d = 1.5;
while (d != 1) {
    std::cout << d << " ";
    d -= 0.1;
}
```

Ausgabe/Output:

2 P

(c)

```
int j;
for (j = 1; j <= 3; ++j)
    std::cout << j;
for (int k = 7-j; k >= 1; --k)
    std::cout << "*";
```

Ausgabe/Output:

2 P

2 Schleifenausgaben (6 Punkte)

Geben Sie für jedes der folgenden drei Code-Stücke die Ausgabe aus, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"! *For each of the following three code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!*

(a)

```
double d = 2.0;
do {
    std::cout << (--d)++;
} while (d < 2.0);
```

Ausgabe/Output:

1

2 P

(b)

```
double d = 1.5;
while (d != 1) {
    std::cout << d << " ";
    d -= 0.1;
}
```

Ausgabe/Output:

infinite loop

2 P

(c)

```
int j;
for (j = 1; j <= 3; ++j)
    std::cout << j;
for (int k = 7-j; k >= 1; --k)
    std::cout << "*";
```

Ausgabe/Output:

123***

2 P

2 Schleifenausgaben (8 Punkte)

Geben Sie für jedes der folgenden vier Code-Stücke die Ausgabe aus, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"! *For each of the following four code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!*

-
- (a)

```
int a[3] = {2, 3, 1};  
for (int i = 0; i < 3; i = a[i])  
    std::cout << i << ' ';
```

Ausgabe/Output:

2 P

-
- (b)

```
unsigned int n = 32;  
do {  
    std::cout << n % 3 << ' ';  
    n /= 2;  
} while (n > 0);
```

Ausgabe/Output:

2 P

-
- (c)

```
double d = 1.0;  
while (d != 0.125) {  
    std::cout << d << ' ';  
    d = d * 0.5;  
}
```

Ausgabe/Output:

2 P

-
- (d)

```
for (int n = 0; n <= 10; n += 2) {  
    if (n % 3 == 0)  
        continue;  
    std::cout << n << ' ';  
}
```

Ausgabe/Output:

2 P

2 Schleifenausgaben (8 Punkte)

Geben Sie für jedes der folgenden vier Code-Stücke die Ausgabe aus, die generiert wird! Nehmen Sie für Fließkommazahlen den Standard IEEE 754 an! Falls es sich um eine Endlosschleife handelt, schreiben Sie "Endlosschleife"! *For each of the following four code snippets, provide the output that is being generated! For floating point numbers, assume the IEEE 754 standard! In case of an infinite loop, write "infinite loop"!*

-
- (a)

```
int a[3] = {2, 3, 1};  
for (int i = 0; i < 3; i = a[i])  
    std::cout << i << ' ';
```

Ausgabe/Output:

0 2 1

2 P

-
- (b)

```
unsigned int n = 32;  
do {  
    std::cout << n % 3 << ' ';  
    n /= 2;  
} while (n > 0);
```

Ausgabe/Output:

2 1 2 1 2 1

2 P

-
- (c)

```
double d = 1.0;  
while (d != 0.125) {  
    std::cout << d << ' ';  
    d = d * 0.5;  
}
```

Ausgabe/Output:

1 0.5 0.25

2 P

-
- (d)

```
for (int n = 0; n <= 10; n += 2) {  
    if (n % 3 == 0)  
        continue;  
    std::cout << n << ' ';  
}
```

Ausgabe/Output:

2 4 8 10

2 P

3 Programmausgaben (10 Punkte)

Betrachten Sie folgende Funktionen und geben Sie die fehlenden Nachbedingungen an. Die Nachbedingungen können informell sein, müssen aber die Ergebnisse und Effekte der Funktionen in Abhängigkeit von den Parametern vollständig charakterisieren.

Consider the following functions and provide the missing post conditions. The post conditions can be reasonably informal, but they must completely characterize the results and effects of the functions depending on the provided parameters.

(a)

2 P

```
// pre: n >= 0
```

```
// post:
```

```
int s(int n){  
    if (n>0)  
        return n+s(n-1);  
    return 0;  
}
```

(b)

2 P

```
// pre: n > 0
```

```
// post:
```

```
int d(int n){  
    int res = 0;  
    while (n > 0){  
        n = n / 10;  
        res++;  
    }  
    return res;  
}
```

(c)

2 P

// pre: $n > 1$

// post:

```
bool p(int n){
    int f=2;
    for (; n%f != 0; ++f);
    return n==f;
}
```

(d)

2 P

```
// pre: An array of characters provided by start pointer start
// and one-past-the-end pointer end
```

// post:

```
void r(char* start, char* end){
    char* it1 = start-1;
    char* it2 = end;
    while (++it1 < --it2){
        char t = *it1;
        *it1 = *it2;
        *it2 = t;
    }
}
```

(e)

2 P

// pre: positive integers a and b

// post:

```
int g(int a, int b){
    if (a == 0) return b;
    return g(b % a, a);
}
```

3 Programmausgaben (10 Punkte)

Betrachten Sie folgende Funktionen und geben Sie die fehlenden Nachbedingungen an. Die Nachbedingungen können informell sein, müssen aber die Ergebnisse und Effekte der Funktionen in Abhängigkeit von den Parametern vollständig charakterisieren.

Consider the following functions and provide the missing post conditions. The post conditions can be reasonably informal, but they must completely characterize the results and effects of the functions depending on the provided parameters.

(a)

2 P

```
// pre: n >= 0  
  
// post: returns the sum of 0 .. n ;  
  
int s(int n){  
    if (n>0)  
        return n+s(n-1);  
    return 0;  
}
```

(b)

2 P

```
// pre: n > 0  
  
// post: return number of digits of n ;  
  
int d(int n){  
    int res = 0;  
    while (n > 0){  
        n = n / 10;  
        res++;  
    }  
    return res;  
}
```

(c)

2 P

```
// pre: n > 1

// post: returns if n is a prime number ;

bool p(int n){
    int f=2;
    for (; n%f != 0; ++f);
    return n==f;
}
```

(d)

2 P

```
// pre: An array of characters provided by start pointer start
// and one-past-the-end pointer end

// post: reverse the string contained in the array from start ..
//        end-1 ;

void r(char* start, char* end){
    char* it1 = start-1;
    char* it2 = end;
    while (++it1 < --it2){
        char t = *it1;
        *it1 = *it2;
        *it2 = t;
    }
}
```

(e)

2 P

```
// pre: positive integers a and b

// post: return greatest common divisor of a and b ;

int g(int a, int b){
    if (a == 0) return b;
    return g(b % a, a);
}
```

6 Lineare Algebra (8 Punkte)

Folgendes Programm deklariert und benutzt eine Klasse für dreidimensionale Vektoren; Vektoren können addiert und mit einem Skalar multipliziert werden. Für zwei Vektoren $v = (v_1, v_2, v_3)$, $w = (w_1, w_2, w_3)$ und eine reelle Zahl s gilt $v + w := (v_1 + w_1, v_2 + w_2, v_3 + w_3)$ und $sv := (sv_1, sv_2, sv_3)$. Ihre Aufgabe ist es, die Funktionen für Skalarmultiplikation und die Vektorausgabe zu definieren, wobei Sie die beiden gegebenen Konstruktoren benutzen dürfen. Beantworten Sie die Fragen auf der rechten Seite!

```
#include<iostream>

struct Vector {
    double x[3];

    // POST: creates the null vector
    Vector ();

    // POST: creates the vector (x1, x2, x3)
    Vector (double x1, double x2, double x3);
};

// POST: returns v + w (vector addition)
Vector operator+ (const Vector& v, const Vector& w);

// POST: returns s * v (scalar multiplication)
Vector operator* (double s, const Vector& v);

// POST: writes v to o in the format (v.x[0], v.x[1], v.x[2])
std::ostream& operator<< (std::ostream& o, const Vector& v);

int main()
{
    Vector v (1, 2, 3);
    Vector w (3, 4, 5);
    std::cout << 2 * v + 3 * w; // (11, 16, 21)
    return 0;
}
```

The program above declares and uses a class for three-dimensional vectors. Vectors can be added and multiplied with a scalar. For two vectors $v = (v_1, v_2, v_3)$, $w = (w_1, w_2, w_3)$ and a real number s , we have $v + w := (v_1 + w_1, v_2 + w_2, v_3 + w_3)$ and $sv := (sv_1, sv_2, sv_3)$. Your task is to define the functions for scalar multiplication, and vector output, where you may use the two given constructors. Answer the questions on the right-hand side!

-
- (a) Definieren Sie die folgende Funktion zur Skalarmultiplikation!

3 P

Define the following function for scalar multiplication!

```
// POST: returns s * v (scalar multiplication)
Vector operator* (double s, const Vector& v) {
```

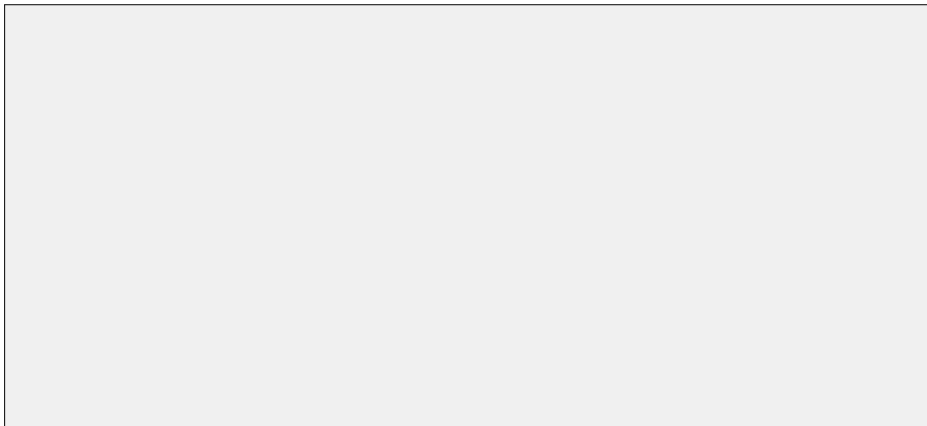


```
}
```

-
- (b) Definieren Sie die folgende Funktion zur Ausgabe eines Vektors! Achten Sie darauf, dass Sie das in der Nachbedingung spezifizierte Format einhalten, das heisst den Vektor wie angegeben mit umschliessenden Klammern und kommaseparierten Elementen ausgeben!
Define the following function for vector output! Make sure that you comply with the format specified in the postcondition, i.e. that you output the vector using surrounding parentheses and comma-separated entries!

5 P

```
// POST: writes v to o in the format (v.x[0], v.x[1], v.x[2])
std::ostream& operator<< (std::ostream& o, const Vector& v) {
```



```
}
```

-
- (a) Definieren Sie die folgende Funktion zur Skalarmultiplikation!
Define the following function for scalar multiplication!

3 P

```
// POST: returns s * v (scalar multiplication)
Vector operator* (double s, const Vector& v) {
```

```
    Vector result;
    for (int i=0; i<3; ++i)
        result.x[i] = s * v.x[i];
    return result;
```

```
}
```

-
- (b) Definieren Sie die folgende Funktion zur Ausgabe eines Vektors! Achten Sie darauf, dass Sie das in der Nachbedingung spezifizierte Format einhalten, das heisst den Vektor wie angegeben mit umschliessenden Klammern und kommaseparierten Elementen ausgeben!
Define the following function for vector output! Make sure that you comply with the format specified in the postcondition, i.e. that you output the vector using surrounding parentheses and comma-separated entries!

5 P

```
// POST: writes v to o in the format (v.x[0], v.x[1], v.x[2])
std::ostream& operator<< (std::ostream& o, const Vector& v) {
```

```
    o << '(';
    for (int i=0; i<3; ++i) {
        o << v.x[i];
        if (i < 2)
            o << ", ";
        else
            o << ')';
    }
    return o;
```

```
}
```

5 Binärdarstellung / Binary representation (6 Punkte)

Berechnen Sie die Binärdarstellungen $b_0.b_{-1}b_{-2}\dots$ der unten angegebenen rationalen Zahlen $q \in (0, 2)$, das heisst, schreiben Sie q jeweils in der Form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0, 1\} \text{ für alle } i.$$

Falls die Darstellung nicht endlich ist, geben Sie die exakte Periode an! Im folgenden finden Sie drei Beispiele.

q	$b_0.b_{-1}b_{-2}\dots$
$3/2$	1.1
$7/8$	0.111
$4/3$	$1.\overline{01}$

Compute the binary representations $b_0.b_{-1}b_{-2}\dots$ of the rational numbers $q \in (0, 2)$ given below, i.e. write q in the form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0, 1\} \text{ for all } i.$$

If the representation is not finite, provide the exact period! Above, you find three examples.

q	$b_0.b_{-1}b_{-2}\dots$
$5/4$	
$31/32$	
$4/7$	
$3/7$	

5 Binärdarstellung / Binary representation (6 Punkte)

Berechnen Sie die Binärdarstellungen $b_0.b_{-1}b_{-2}\dots$ der unten angegebenen rationalen Zahlen $q \in (0, 2)$, das heisst, schreiben Sie q jeweils in der Form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0, 1\} \text{ für alle } i.$$

Falls die Darstellung nicht endlich ist, geben Sie die exakte Periode an! Im folgenden finden Sie drei Beispiele.

q	$b_0.b_{-1}b_{-2}\dots$
$3/2$	1.1
$7/8$	0.111
$4/3$	$1.\overline{01}$

Compute the binary representations $b_0.b_{-1}b_{-2}\dots$ of the rational numbers $q \in (0, 2)$ given below, i.e. write q in the form

$$q = \sum_{i=0}^{\infty} b_{-i}2^{-i}, \quad b_{-i} \in \{0, 1\} \text{ for all } i.$$

If the representation is not finite, provide the exact period! Above, you find three examples.

q	$b_0.b_{-1}b_{-2}\dots$
$5/4$	1.01
$31/32$	0.11111
$4/7$	$0.\overline{100}$
$3/7$	$0.\overline{011}$

5 Normalisierte Fließkommazahlen (8 Punkte)

Betrachten Sie die Funktion *Consider the function*

$$f(a, b, c, e) = (1 + a \cdot 2^{-1} + b \cdot 2^{-2} + c \cdot 2^{-3}) \cdot 2^e,$$

$$a, b, c \in \{0, 1\}, \quad e \in \mathbb{Z}.$$

Geben Sie in den folgenden Gleichungen jeweils Werte für die Argumente a, b, c und e so an, dass der Funktionswert die angegebene reelle Zahl ist!

In each of the following equalities, supply values for the arguments a, b, c and e such that the function value is the required real number!

$$(a) \left(1 + \boxed{} \cdot 2^{-1} + \boxed{} \cdot 2^{-2} + \boxed{} \cdot 2^{-3} \right) \cdot 2^{\boxed{}} = 2$$

$$(b) \left(1 + \boxed{} \cdot 2^{-1} + \boxed{} \cdot 2^{-2} + \boxed{} \cdot 2^{-3} \right) \cdot 2^{\boxed{}} = \frac{7}{8}$$

$$(c) \left(1 + \boxed{} \cdot 2^{-1} + \boxed{} \cdot 2^{-2} + \boxed{} \cdot 2^{-3} \right) \cdot 2^{\boxed{}} = 30$$

$$(d) \left(1 + \boxed{} \cdot 2^{-1} + \boxed{} \cdot 2^{-2} + \boxed{} \cdot 2^{-3} \right) \cdot 2^{\boxed{}} = 4.5$$

5 Normalisierte Fließkommazahlen (8 Punkte)

Betrachten Sie die Funktion *Consider the function*

$$f(a, b, c, e) = (1 + a \cdot 2^{-1} + b \cdot 2^{-2} + c \cdot 2^{-3}) \cdot 2^e,$$

$$a, b, c \in \{0, 1\}, \quad e \in \mathbb{Z}.$$

Geben Sie in den folgenden Gleichungen jeweils Werte für die Argumente a, b, c und e so an, dass der Funktionswert die angegebene reelle Zahl ist!

In each of the following equalities, supply values for the arguments a, b, c and e such that the function value is the required real number!

$$(a) \left(1 + \boxed{0} \cdot 2^{-1} + \boxed{0} \cdot 2^{-2} + \boxed{0} \cdot 2^{-3} \right) \cdot 2^{\boxed{1}} = 2$$

$$(b) \left(1 + \boxed{1} \cdot 2^{-1} + \boxed{1} \cdot 2^{-2} + \boxed{0} \cdot 2^{-3} \right) \cdot 2^{\boxed{-1}} = \frac{7}{8}$$

$$(c) \left(1 + \boxed{1} \cdot 2^{-1} + \boxed{1} \cdot 2^{-2} + \boxed{1} \cdot 2^{-3} \right) \cdot 2^{\boxed{4}} = 30$$

$$(d) \left(1 + \boxed{0} \cdot 2^{-1} + \boxed{0} \cdot 2^{-2} + \boxed{1} \cdot 2^{-3} \right) \cdot 2^{\boxed{2}} = 4.5$$

5 Normalisierte Fließkommazahlen (8 Punkte)

Wir betrachten das unten angegebene normalisierte Fließkommazahlensystem F^* . Beantworten Sie die Fragen auf der rechten Seite!

Anmerkung: Falls nötig, runden Sie wie folgt: bei einer 1 direkt hinter der letzten signifikanten Stelle wird aufgerundet, bei einer 0 wird abgerundet.

Beispiel: in F^* wird die binär dargestellte Zahl $1.01\underline{0}..$ zu 1.01 abgerundet, während $1.01\underline{1}..$ zu 1.10 aufgerundet wird.

$F^*(\beta, p, e_{\min}, e_{\max})$ mit / *with*

$$\beta = 2$$

$$p = 3$$

$$e_{\min} = -3$$

$$e_{\max} = 3$$

Consider the normalized floating point number system F^ as defined above. Answer the questions on the right hand side!*

Remark: *If necessary, use the following rounding mode: if there is a 1 directly behind the last significant digit, round up, and if there is a 0, round down.*

Example: *in F^* , the binary represented number $1.01\underline{0}..$ is rounded down to 1.01, while $1.01\underline{1}..$ is rounded up to 1.10.*

- (a) Wie viele unterschiedliche positive Werte enthält das normalisierte Fließkommazahlensystem F^* ? 1 P

How many different positive values does the normalized floating point number system F^ contain?*

- (b) Geben Sie die grösste Zahl und die kleinste positive Zahl an, die im normalisierten Fließkommazahlensystem F^* enthalten ist. Beide Antworten sind in **dezimaler Darstellung** anzugeben. 2 P

Provide the largest number and the smallest positive number contained in the normalized floating point number system F^ . Provide both answers in **decimal representation**.*

grösste Zahl / *largest number*

kleinste positive Zahl / *smallest positive number*

- (c) Berechnen Sie folgende Ausdrücke so wie sie geklammert sind, indem Sie gemäss den Regeln für das Rechnen mit Fließkommazahlen jedes Zwischenresultat (und das Endresultat) im gegebenen Fließkommazahlensystem darstellen. Vergessen Sie das korrekte Runden nicht (siehe Anmerkung auf der linken Seite)! 5 P

Compute the following expressions as the parentheses suggest, representing each intermediate result (and the final result) in the given normalized floating point number system according to the rules of computing with floating point numbers. Do not forget to round correctly (see Remark on the left hand side)!

$$(0.75 + 0.5) + 10$$

$$(10 + 0.75) + 0.5$$

dezimal <i>decimal</i>	binär <i>binary</i>	dezimal <i>decimal</i>	binär / <i>binary</i>
0.75	<input style="width: 100%; height: 30px;" type="text"/>	10	<input style="width: 100%; height: 30px;" type="text"/>
+ 0.5	<input style="width: 100%; height: 30px;" type="text"/>	+ 0.75	<input style="width: 100%; height: 30px;" type="text"/>
=	<input style="width: 100%; height: 30px;" type="text"/>	=	<input style="width: 100%; height: 30px;" type="text"/>
+ 10	<input style="width: 100%; height: 30px;" type="text"/>	+ 0.5	<input style="width: 100%; height: 30px;" type="text"/>
= <input style="width: 50px; height: 30px;" type="text"/> ←	<input style="width: 100%; height: 30px;" type="text"/>	= <input style="width: 50px; height: 30px;" type="text"/> ←	<input style="width: 100%; height: 30px;" type="text"/>

- (a) Wie viele unterschiedliche positive Werte enthält das normalisierte Fließkommazahlensystem F^* ? 1 P

How many different positive values does the normalized floating point number system F^ contain?*

28

- (b) Geben Sie die grösste Zahl und die kleinste positive Zahl an, die im normalisierten Fließkommazahlensystem F^* enthalten ist. Beide Antworten sind in **dezimaler Darstellung** anzugeben. 2 P

Provide the largest number and the smallest positive number contained in the normalized floating point number system F^ . Provide both answers in **decimal representation**.*

grösste Zahl / *largest number*

14

kleinste positive Zahl / *smallest positive number*

1/8

- (c) Berechnen Sie folgende Ausdrücke so wie sie geklammert sind, indem Sie gemäss den Regeln für das Rechnen mit Fließkommazahlen jedes Zwischenresultat (und das Endresultat) im gegebenen Fließkommazahlensystem darstellen. Vergessen Sie das korrekte Runden nicht (siehe Anmerkung auf der linken Seite)! 5 P

Compute the following expressions as the parentheses suggest, representing each intermediate result (and the final result) in the given normalized floating point number system according to the rules of computing with floating point numbers. Do not forget to round correctly (see Remark on the left hand side)!

$$(0.75 + 0.5) + 10$$

$$(10 + 0.75) + 0.5$$

dezimal <i>decimal</i>	binär <i>binary</i>	dezimal <i>decimal</i>	binär / <i>binary</i>
0.75	$1.10 \cdot 2^{-1}$	10	$1.01 \cdot 2^3$
+ 0.5	$1.00 \cdot 2^{-1}$	+ 0.75	$1.10 \cdot 2^{-1}$
=	$1.01 \cdot 2^0$	=	$1.01 \cdot 2^3$
+ 10	$1.01 \cdot 2^3$	+ 0.5	$1.00 \cdot 2^{-1}$
= 12 ←	$1.10 \cdot 2^3$	= 10 ←	$1.01 \cdot 2^3$

Aufgabe 4: EBNF I (6P)

Die folgende EBNF definiert erlaubte Anweisungen einer vereinfachten Programmiersprache.

Beantworten Sie die Fragen auf der rechten Seite.

Anmerkung: Leerschläge sind im Rahmen der EBNF bedeutungslos.

The following EBNF defines the allowed statements of a programming language. Answer the questions on the right hand side.

Remark: *Whitespaces are irrelevant in the context of this EBNF.*

Statement	= Expression ';'.
Expression	= Simple ':' Simple.
Simple	= Designator Integer.
Designator	= Identifier { '.' Identifier '(' [ExpressionList] ')' }.
ExpressionList	= Expression { ',' Expression }.
Integer	= Digit {Digit}.
Digit	= '0' '1' '2' '3' '4' '5' '6' '7' '8' '9' .
Identifier	= Letter {Letter}.
Letter	= 'a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' .

- (a) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

$a(2:5);$

☐ wahr / *true* ☐ falsch / *false*

- (b) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

$a(b).c:d;$

☐ wahr / *true* ☐ falsch / *false*

- (c) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

$a(3).3;$

☐ wahr / *true* ☐ falsch / *false*

- (d) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

$a(3):3;$

☐ wahr / *true* ☐ falsch / *false*

- (e) Ändern Sie genau eine Produktionsregel der EBNF ab, so dass folgende Anweisung (statement) gültig wird. *Modify one and only one production rule of the EBNF such that the following statement becomes valid.* /2P

$a3(c3):a4(c4);$

Geänderte Zeile / *Modified line:*

Aufgabe 5: EBNF II (8P)

Auf der nächsten Seite ist Code abgebildet, mit welchem identifiziert werden soll, ob eine Zeichenkette eine im Sinne obiger EBNF gültige Anweisung (Statement) darstellt. Folgender Code, welcher nur als Funktionsdeklaration vorliegt, soll als korrekt implementiert vorausgesetzt werden.

Consider the code on the next page that shall be used to check if a string provides a valid statement corresponding to the EBNF above. The following code that is only provided as a function declaration can be considered correctly implemented.

```
// POST: when the next available non-whitespace character equals c,  
//       it is consumed and the function returns true,  
//       otherwise the function returns false.  
bool has(std::istream& is, char c);  
  
// Integer = Digit {Digit}.  
bool Integer (std::istream& is);  
  
// Identifier = Letter {Letter}.  
bool Identifier (std::istream& is);
```

- /2P (a) Die Funktion Expression wird im Code anscheinend zweimal deklariert. Erklären Sie warum.

The function Expression seems to be declared twice in the code. Explain why.

- /2P (b) Ergänzen Sie die Funktion Designator, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function Designator such that it implements the corresponding EBNF line correctly.

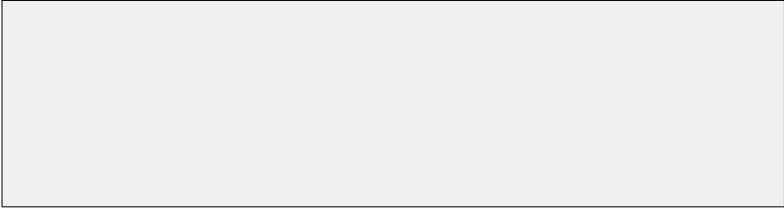
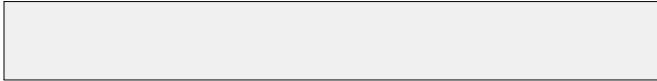

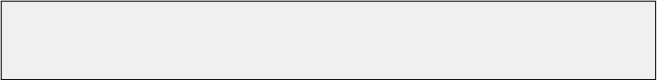
- /2P (c) Ergänzen Sie die Funktion ExpressionList, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function ExpressionList such that it implements the corresponding EBNF line correctly.

- /2P (d) Ergänzen Sie die Funktion Expression, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function Expression such that it implements the corresponding EBNF line correctly.


```

bool Expression (std::istream& is);
// ExpressionList = Expression { ',' Expression }.
bool ExpressionList(std::istream& is){
    if (!Expression(is)) return false;
    
    return true;
}
// Designator = Identifier { '.' Identifier | '(' [ExpressionList] ')' }.
bool Designator(std::istream& is){
    if (!Identifier(is)) return false;
    while(true){
        if (has(is, '.')){
            ;
        } else if (has(is, '(')){
            bool ignore = ExpressionList(is);
            ;
        } else
            return true;
    }
}
// Simple = Designator | Integer.
bool Simple(std::istream& is){
    return Integer(is) || Designator(is);
}
// Expression = Simple [ ':' Simple ].
bool Expression(std::istream& is){
    if (!Simple(is)) return false;
    return ;
}
// Statement = Expression ';' .
bool Statement(std::istream& is){
    return Expression(is) && has(is, ';');
}

```

Aufgabe 4: EBNF I (6P)

Die folgende EBNF definiert erlaubte Anweisungen einer vereinfachten Programmiersprache.

Beantworten Sie die Fragen auf der rechten Seite.

Anmerkung: Leerschläge sind im Rahmen der EBNF bedeutungslos.

The following EBNF defines the allowed statements of a programming language. Answer the questions on the right hand side.

Remark: *Whitespaces are irrelevant in the context of this EBNF.*

Statement	= Expression ';'.
Expression	= Simple ':' Simple.
Simple	= Designator Integer.
Designator	= Identifier { '.' Identifier '(' [ExpressionList] ')' }.
ExpressionList	= Expression { ',' Expression }.
Integer	= Digit {Digit}.
Digit	= '0' '1' '2' '3' '4' '5' '6' '7' '8' '9' .
Identifier	= Letter {Letter}.
Letter	= 'a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' .

- (a) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

a(2:5);

☒ wahr / true ☐ falsch / false

- (b) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

a(b).c:d;

☒ wahr / true ☐ falsch / false

- (c) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

a(3).3;

☐ wahr / true ☒ falsch / false

- (d) Folgende Zeichenkette entspricht einer gültigen Anweisung (statement) gemäss der EBNF: *The following string corresponds to a valid statement according to the EBNF:* /1P

a(3):3;

☒ wahr / true ☐ falsch / false

- (e) Ändern Sie genau eine Produktionsregel der EBNF ab, so dass folgende Anweisung (statement) gültig wird. *Modify one and only one production rule of the EBNF such that the following statement becomes valid.* /2P

a3(c3):a4(c4);

Geänderte Zeile / *Modified line:*

Identifizier = Letter {Letter | Digit}.

Aufgabe 5: EBNF II (8P)

Auf der nächsten Seite ist Code abgebildet, mit welchem identifiziert werden soll, ob eine Zeichenkette eine im Sinne obiger EBNF gültige Anweisung (Statement) darstellt. Folgender Code, welcher nur als Funktionsdeklaration vorliegt, soll als korrekt implementiert vorausgesetzt werden.

Consider the code on the next page that shall be used to check if a string provides a valid statement corresponding to the EBNF above. The following code that is only provided as a function declaration can be considered correctly implemented.

```
// POST: when the next available non-whitespace character equals c,
//       it is consumed and the function returns true,
//       otherwise the function returns false.
bool has(std::istream& is, char c);

// Integer = Digit {Digit}.
bool Integer (std::istream& is);

// Identifier = Letter {Letter}.
bool Identifier (std::istream& is);
```

- /2P (a) Die Funktion Expression wird im Code anscheinend zweimal deklariert. Erklären Sie warum.

The function Expression seems to be declared twice in the code. Explain why.

Es gibt eine zirkuläre Abhängigkeit zwischen den Funktionen. Daher muss mindestens eine Funktion vor ihrer Definition deklariert werden. Sonst wäre sie in mindestens einer anderen Funktion nicht sichtbar.

- /2P (b) Ergänzen Sie die Funktion Designator, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function Designator such that it implements the corresponding EBNF line correctly.

- /2P (c) Ergänzen Sie die Funktion ExpressionList, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function ExpressionList such that it implements the corresponding EBNF line correctly.

- /2P (d) Ergänzen Sie die Funktion Expression, so dass Sie die entsprechende EBNF-Zeile korrekt implementiert.

Complement function Expression such that it implements the corresponding EBNF line correctly.

```

bool Expression (std::istream& is);
// ExpressionList = Expression { ',' Expression }.
bool ExpressionList(std::istream& is){
    if (!Expression(is)) return false;

    while (has(is, ',')){
        if (!Expression(is)) return false;
    }

    return true;
}

// Designator = Identifier { '.' Identifier | '(' [ExpressionList] ')' }.
bool Designator(std::istream& is){
    if (!Identifier(is)) return false;
    while(true){
        if (has(is, '.')){
            if (!Identifier(is)) return false;
        } else if (has(is, '(')){
            bool ignore = ExpressionList(is);
            if (!has(is, ')')) return false;
        } else
            return true;
    }
}

// Simple = Designator | Integer.
bool Simple(std::istream& is){
    return Integer(is) || Designator(is);
}

// Expression = Simple [ ':' Simple ].
bool Expression(std::istream& is){
    if (!Simple(is)) return false;

    return !has(is, ':') || Simple(is);
}

// Statement = Expression ';' .
bool Statement(std::istream& is){
    return Expression(is) && has(is, ';');
}

```

6 Arithmetic Stack (6 Punkte)

Folgender Code implementiert einen Teil eines sogenannten arithmetischen Stacks. Beantworten Sie die Fragen auf der rechten Seite.

```
#include <iostream>
class arithmetic_stack{
    double stack[8]; // stack values
    int position;
public:
    arithmetic_stack() {
        position = 0;
    }
    // PRE: position < 8; post: pushes value to new stack top
    void push(double value) {
        stack[position++] = value;
    }
    // PRE: position > 0; post: pops and returns value at stack top
    double pop() {
        return stack[--position];
    }
    // PRE: position > 1; post: replaces two top-most stack values by the sum
    void add() {
        push(pop() + pop());
    }
    // PRE: position > 1; post: replaces two top-most stack values by the product
    void mul() {
        push(pop() * pop());
    }
}

int main() {
    arithmetic_stack stack;
    stack.push(30); stack.push(20); stack.push(10);
    stack.add(); stack.mul();
    std::cout << stack.pop();
    return 0;
}
```

The code above implements a part of a so called arithmetic stack. Answer the questions on the right-hand side.

- (a) Was gibt die Mainfunktion aus?

1 P

What is the output of the main function?

Antwort / answer:

- (b) Angenommen, a, b und c seien Variablen vom Typ double. Streichen Sie in den folgenden Codefragmenten zu entfernende Anweisungen in grauen Boxen durch, so dass der angegebene Ausdruck berechnet wird.

2 P

Assume a, b and c are variables of type double. In the following code fragments, strike through statements to be removed in grey boxes such that the provided expressions are computed.

```
// compute a * b + c
arithmetic_stack s;
s.push(a); s.push(b);


s.mul();



s.add();


s.push(c);


s.mul();



s.add();


std::cout << s.pop();
```

```
// compute a + b * c
arithmetic_stack s;
s.push(a); s.push(b);


s.mul();



s.add();


s.push(c);


s.mul();



s.add();


std::cout << s.pop();
```

- (c) Vervollständigen Sie folgende Funktion so, dass sie das Skalarprodukt
- $\sum_{i=0}^{len-1} a_i \cdot b_i$
- von
- $a \in \mathbb{R}^{len}$
- und
- $b \in \mathbb{R}^{len}$
- zurückgibt.

3 P

Complete the following function such that it returns the scalar product $\sum_{i=0}^{len-1} a_i \cdot b_i$ of $a \in \mathbb{R}^{len}$ and $b \in \mathbb{R}^{len}$.

```
double scalarproduct(double a[], double b[], int len) {
    arithmetic_stack stack;
    ;
    for (int i = 0; i < len; ++i){
        stack.push(a[i]); stack.push(b[i]);
        ;
        ;
    }
    return stack.pop();
}
```

6 Arithmetic Stack (6 Punkte)

Folgender Code implementiert einen Teil eines sogenannten arithmetischen Stacks. Beantworten Sie die Fragen auf der rechten Seite.

```
#include <iostream>
class arithmetic_stack{
    double stack[8]; // stack values
    int position;
public:
    arithmetic_stack() {
        position = 0;
    }
    // PRE: position < 8; post: pushes value to new stack top
    void push(double value) {
        stack[position++] = value;
    }
    // PRE: position > 0; post: pops and returns value at stack top
    double pop() {
        return stack[--position];
    }
    // PRE: position > 1; post: replaces two top-most stack values by the sum
    void add() {
        push(pop() + pop());
    }
    // PRE: position > 1; post: replaces two top-most stack values by the product
    void mul() {
        push(pop() * pop());
    }
}

int main() {
    arithmetic_stack stack;
    stack.push(30); stack.push(20); stack.push(10);
    stack.add(); stack.mul();
    std::cout << stack.pop();
    return 0;
}
```

The code above implements a part of a so called arithmetic stack. Answer the questions on the right-hand side.

- (a) Was gibt die Mainfunktion aus?

1 P

What is the output of the main function?

Antwort / answer: 900

- (b) Angenommen, a, b und c seien Variablen vom Typ double. Streichen Sie in den folgenden Codefragmenten zu entfernende Anweisungen in grauen Boxen durch, so dass der angegebene Ausdruck berechnet wird.

2 P

Assume a, b and c are variables of type double. In the following code fragments, strike through statements to be removed in grey boxes such that the provided expressions are computed.

```
// compute a * b + c
arithmetic_stack s;
s.push(a); s.push(b);


s.mul();



s.add();


s.push(c);


s.mul();



s.add();


std::cout << s.pop();
```

```
// compute a + b * c
arithmetic_stack s;
s.push(a); s.push(b);


s.mul();



s.add();


s.push(c);


s.mul();



s.add();


std::cout << s.pop();
```

- (c) Vervollständigen Sie folgende Funktion so, dass sie das Skalarprodukt
- $\sum_{i=0}^{len-1} a_i \cdot b_i$
- von
- $a \in \mathbb{R}^{len}$
- und
- $b \in \mathbb{R}^{len}$
- zurückgibt.

3 P

Complete the following function such that it returns the scalar product $\sum_{i=0}^{len-1} a_i \cdot b_i$ of $a \in \mathbb{R}^{len}$ and $b \in \mathbb{R}^{len}$.

```
double scalarproduct(double a[], double b[], int len) {
    arithmetic_stack stack;
    

stack.push(0)

;
    for (int i = 0; i < len; ++i){
        stack.push(a[i]); stack.push(b[i]);
        

stack.mul()

;
        

stack.add()

;
    }
    return stack.pop();
}
```

7 Prüfsummenberechnung (6 Punkte)

Eine Prüfsumme wird aus einer Reihe von Werten gebildet und wird verwendet um deren Integrität zu überprüfen. Vervollständigen Sie folgende Klasse, welche die Funktionalität zur Berechnung einer sehr einfachen Prüfsumme natürlicher Zahlen kapselt.

1. Konstruktor Checksum: initialisiert `sum` mit 0.
2. Member-Funktion `add`: Addiert eine nichtnegative ganze Zahl zu `sum`.
3. Member-Funktion `get`: Gibt den Wert `sum` der berechneten Prüfsumme zurück. `sum` wird nicht geändert.

```
class Checksum {
    D1 unsigned int sum;
    A :
    Checksum() : B {}

    void add(D2 unsigned int value) D3 {
        C ;
    }
    unsigned int get() D4 {
        return E ;
    }
};
```

A checksum is generated out of a series of values and is used to verify the integrity of these values. Complete the class `Checksum` that implements functionality to calculate a very simple checksum of natural numbers.

1. Constructor `Checksum`: initializes member variable `sum` with 0.
2. Member function `add`: Adds an unsigned int value to the member variable `sum`.
3. Member function `get`: Computes and returns the checksum `sum`. It does not modify `sum`.

-
- (a) Welche Deklaration muss bei A eingesetzt werden, so dass folgendes Programm kompiliert? 1 P
Fill in the declaration in A, to make the following program compile.

```
#include <iostream>
int main() {
    Checksum cs;
    cs.add (3); cs.add (4); cs.add (5);
    std::cout << cs.get() << "\n"; // 12
}
```

A:

-
- (b) Welche Initialisierungsanweisung muss bei B eingesetzt werden? 1 P
Fill in the initialisation statement for B.

B:

-
- (c) Welche Anweisung muss bei C eingesetzt werden? 1 P
Fill in the statement for C.

C:

-
- (d) Betrachten Sie Markierungen D1, D2, D3 und D4. Nennen Sie diejenigen Markierungen, an denen das Schlüsselwort `const` eingesetzt werden kann, so dass die Klasse noch kompiliert werden kann. 2 P
Look at the markings D1,D2,D3,D4. Name those markings at which the keyword `const` can be inserted such that the class can still be compiled.

D:

-
- (e) Welcher Ausdruck muss für E eingesetzt werden? 1 P
Which expression must be inserted for E?

E:

7 Prüfsummenberechnung (6 Punkte)

Eine Prüfsumme wird aus einer Reihe von Werten gebildet und wird verwendet um deren Integrität zu überprüfen. Vervollständigen Sie folgende Klasse, welche die Funktionalität zur Berechnung einer sehr einfachen Prüfsumme natürlicher Zahlen kapselt.

1. Konstruktor Checksum: initialisiert `sum` mit 0.
2. Member-Funktion `add`: Addiert eine nichtnegative ganze Zahl zu `sum`.
3. Member-Funktion `get`: Gibt den Wert `sum` der berechneten Prüfsumme zurück. `sum` wird nicht geändert.

```
class Checksum {
    D1 unsigned int sum;
    A :
    Checksum() : B {}

    void add(D2 unsigned int value) D3 {
        C ;
    }
    unsigned int get() D4 {
        return E ;
    }
};
```

A checksum is generated out of a series of values and is used to verify the integrity of these values. Complete the class `Checksum` that implements functionality to calculate a very simple checksum of natural numbers.

1. Constructor `Checksum`: initializes member variable `sum` with 0.
2. Member function `add`: Adds an unsigned int value to the member variable `sum`.
3. Member function `get`: Computes and returns the checksum `sum`. It does not modify `sum`.

-
- (a) Welche Deklaration muss bei A eingesetzt werden, so dass folgendes Programm kompiliert? 1 P
Fill in the declaration in A, to make the following program compile.

```
#include <iostream>
int main() {
    Checksum cs;
    cs.add (3); cs.add (4); cs.add (5);
    std::cout << cs.get() << "\n"; // 12
}
```

A: `public`

-
- (b) Welche Initialisierungsanweisung muss bei B eingesetzt werden? 1 P
Fill in the initialisation statement for B.

B: `sum(0)`

-
- (c) Welche Anweisung muss bei C eingesetzt werden? 1 P
Fill in the statement for C.

C: `sum += value`

-
- (d) Betrachten Sie Markierungen D1, D2, D3 und D4. Nennen Sie diejenigen Markierungen, an denen das Schlüsselwort `const` eingesetzt werden kann, so dass die Klasse noch kompiliert werden kann. 2 P
Look at the markings D1,D2,D3,D4. Name those markings at which the keyword `const` can be inserted such that the class can still be compiled.

D: `D2 D4`

-
- (e) Welcher Ausdruck muss für E eingesetzt werden? 1 P
Which expression must be inserted for E?

E: `sum`