

## Aufgabe 1: Typen und Werte (Basistypen) (12P)

Geben Sie für jeden der Ausdrücke auf der nächsten Seite jeweils C++ Typ und Wert an. Wenn der Wert nicht bestimmt werden kann, schreiben Sie "undefiniert".

In manchen Teilaufgaben finden Sie den Ausdruck unter einer dünnen Linie. Darueber finden Sie eine oder mehrere Anweisungen, welche direkt vor dem Ausdruck ausgeführt wurden und welche relevant sind zur Bestimmung des Typs und Wertes des gefragten Ausdrucks.

*For each of the expressions on the next page provide the C++ type and value. If a value cannot be determined, write "undefined".*

*In some tasks you find the expression below a thin line. Above the line you find one or more statements that have been executed immediately before the expression. They are relevant to determine the type and value of the expression in question.*

(a) `4 <= 5.f`

/2P

Typ / *Type*

`bool`

Wert / *Value*

`true`

(b) `2 / 4.0f + 7 / 2.0`

/2P

Typ / *Type*

`double`

Wert / *Value*

`4`

(c) `int a = 4;`  
`int d = 5;`

/2P

`a / d--`

Typ / *Type*

`int`

Wert / *Value*

`0`

(d) `std::vector<float> f = { 1, 2, 3, 4, };`

/2P

`f[2] + (int)f[3]`

Typ / *Type*

`float`

Wert / *Value*

`7`

(e) `int b = 3;`  
`int k = b++;`

/2P

`++k`

Typ / *Type*

`int`

Wert / *Value*

`4`

(f) `(2u - 8 % 5) < 0`

/2P

Typ / *Type*

`bool`

Wert / *Value*

`false`

## Aufgabe 2: Konstrukte (16P)

Geben Sie zu folgenden Codestücken jeweils die erzeugte Ausgabe an.

*Provide the output for each of the following pieces of code.*

```
int a[6] = {1, 3, 5, 7, 9, 11};
int* x = a;
int& k = *x;
x++;
std::cout << *(x+k);
```

/4P (a)

Ausgabe / *Output: 5*

```
float p[6] = {0.5, 1.0, 1.5, 2.0, 0.5, 0.0};
float* s = &p[4];
float* r = &p[2];
std::cout << (r[2] + *(s+1))
```

/4P (b)

Ausgabe / *Output: 0.5*

```
void my_function(int* m) {
    *m = 2;
}
int values[] = {1, 3, 5};
int* v = values;
my_function(v++);
std::cout << v[0];
```

/4P (c)

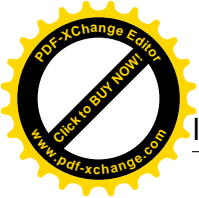
Ausgabe / *Output: 3*

```
struct Node {
    Node* next;
    int value;
    Node (int v, Node* n) : next(n), value(v) {};
};

Node s = Node(10, nullptr);
Node t = Node(20, &s);
s.next = &t;
std::cout << (s.next->next->value);
```

/4P (d)

Ausgabe / *Output: 10*



### Aufgabe 3: Zahlendarstellungen (9P)

- (a) Markieren Sie die Werte, die eine exakte Darstellung in einem beliebigen (endlichen) normalisierten binären Fließkommazahlensystem besitzen.

*Mark the values that have an exact representation in an arbitrary (finite) normalized binary floating point number system.*

/3P

☐ 0.150    ☒ 1.625    ☐ 0.4

- (b) Die Zahl in der linken Spalte der nachfolgenden Tabelle ist jeweils als Literal der Sprache C++ zu verstehen. Berechnen Sie, was jeweils verlangt ist.

*The number displayed to the left in the following table shall be considered a number literal in C++ language. Compute what is requested.*

/3P

0x2A	als Dezimalzahl / <i>to decimal number</i>	= 42
110101	die binäre Zahl (6 Bits, Zweierkomplement) als Dezimalzahl / <i>from binary number with 6 bits in two's complement to decimal number</i>	= -11
-17	die Dezimalzahl als binäre Zahl (7 Bits, Zweierkomplement) / <i>from decimal number to binary number with 7 bits in two's complement</i>	= 1101111

- (c) Geben Sie ein möglichst knappes normalisiertes Fließkommazahlensystem an, mit welchem sich die folgenden dezimalen Werte gerade noch genau darstellen lassen: jede Verkleinerung von  $p$ ,  $e_{\max}$  oder  $-e_{\min}$  muss dazu führen, dass mindestens eine Zahl nicht mehr dargestellt werden kann.

*Provide the smallest possible normalized floating point number system that can still represent the following values exactly: any decrease of the numbers  $p$ ,  $e_{\max}$  or  $-e_{\min}$  must imply that at least one of the numbers cannot be represented any more.*

/3P

Hinweis:  $p$  zählt auch die führende Ziffer.

*Hint:  $p$  does also count the leading digit.*

Tipp: Schreiben Sie sich die normalisierte Binärzahldarstellung der Werte auf, wenn sie für Sie nicht offensichtlich ist.

*Tip: Write down the normalized binary representation of the values, if it is not obvious for you.*

Werte / *Values*: 10, 1.625

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

$\beta = 2$  ,  $p = 4$  ,  $e_{\min} = 0$  ,  $e_{\max} = 3$  .

## Aufgabe 4: EBNF: Ein kleiner Parser (12P)

Die folgende EBNF definiert erlaubte Anweisungen einer vereinfachten Programmiersprache. Sie können annehmen, dass die ersten drei gegebenen Funktionsdeklarationen korrekt implementiert sind. Vervollständigen Sie die Funktionen in den Teilaufgaben.

**Anmerkung:** Leerschläge sind im Rahmen dieser EBNF bedeutungslos.

*The following EBNF defines the allowed expressions in a programming language. The first three function declarations can be considered correctly implemented. Complete the functions in the subparts of this task.*

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

---

```

Expression    = Unit { '.' Unit }.
Unit          = Identifier { '[' Range ']' }.
Range         = Expression | Integer ':' Integer.

Integer       = Digit { Digit }.
Digit         = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' .

Identifier    = Letter { Letter | Digit }.
Letter        = 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' .

```

---

```

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

```

```

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

```

```

// Consumes next Identifier = Letter {Letter | Digit}
bool Identifier (std::istream& is);

```

- 4P (a) Vervollständigen Sie die Funktion Expression gemäss der gegebenen EBNF.

*Complete the function Expression to implement the corresponding EBNF correctly.*

```

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

    while (consume(is, '.')){
        if (!Unit(is)) return false;
    }

    return true;
}

```

- (b) Vervollständigen Sie die Funktion Unit gemäss der gegebenen EBNF.

*Complete the function Unit to implement the corresponding EBNF correctly.*

/4P

```
// Consumes next Unit = Identifier {'[' Range ']'}
```

```
bool Unit(std::istream& is) {
    if (!Identifier(is)) return false;
    while(true) {
        if (consume(is, '[')) {
            if (!Range(is) || !consume(is, ']')) return false;
        } else
            return true;
    }
}
```

- (c) Vervollständigen Sie die Funktion Range gemäss der gegebenen EBNF.

*Complete the function Range to implement the corresponding EBNF correctly.*

/4P

```
//Consumes next Range = Expression | Integer ':' Integer
```

```
bool Range(std::istream& is) {
    if (Expression(is)) return true;
    else if (Integer(is)) {
        return consume(is, ':') && Integer(is);
    }
    return false;
}
```

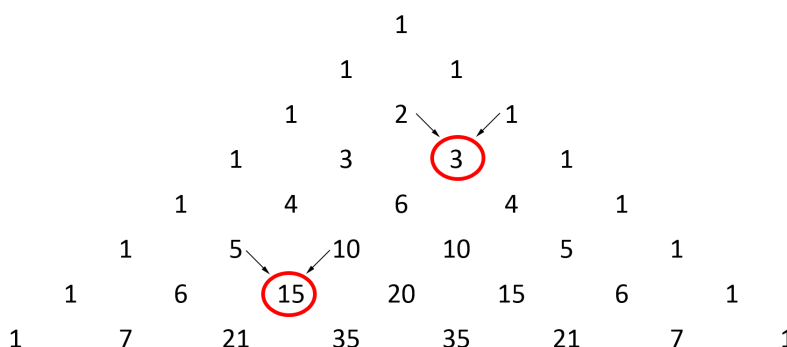
## Aufgabe 5: Rekursion: Pascalsches Dreieck (13P)

Das Ziel dieser Aufgabe ist, die Zahl einer gegebenen Position im Pascalschen Dreieck zu berechnen (siehe Bild). Vervollständigen Sie das gegebene Programm entsprechend.

Eine Zahl an einer gegebenen Position innerhalb des Dreiecks entspricht der Summe der beiden Zahlen links und rechts in der darüberliegenden Zeile. Alle Zahlen am linken und rechten Rand des Dreiecks haben den Wert 1 gesetzt.

Beispiel: Die eingekreiste 15 im untenstehenden Dreieck steht in der 7<sup>ten</sup> Zeile, an 3<sup>te</sup> Stelle. Gleichermassen: Die eingekreiste 3 steht in der 4<sup>ten</sup> Zeile, an 3<sup>ter</sup> Stelle.

Das Programm soll den Nutzer auffordern, eine Zeile und die Stelle in dieser Zeile einzugeben. Dabei wird **mit 1 angefangen** zu zählen. Ausserdem soll das Programm vor der Berechnung überprüfen, ob die eingegebene Positionskombination gültig ist.



*In this task, you need to complete the program below to compute the value of a given position in Pascal's triangle (see image).*

*A number in a given position within the triangle corresponds to the sum of the two numbers on the left and right of the row above. All numbers on the left and right edge of the triangle are set to have value 1.*

*Example: The encircled 15 in the below triangle corresponds to the 7<sup>th</sup> row, on the 3<sup>rd</sup> position. Similarly, encircled 3 corresponds to the 4<sup>th</sup> row, on the 3<sup>rd</sup> position.*

*The program should prompt the user to input a row, and a position in that row, considering **1-based** counting.*

*The program should ensure that the input is valid before computing a value for the position.*

/3P (a) Vervollständigen Sie die Funktion main.

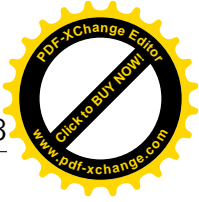
*Complete the function main.*

/5P (b) Vervollständigen Sie die Funktion check\_input\_validity. Sie können annehmen, dass die eingegebenen Variablen immer vom richtigen Datentyp sind.

*Complete the function check\_input\_validity. You can assume that the user always enters integer values as the input variables.*

/5P (c) Vervollständigen Sie die Funktion compute\_pascal mit Hilfe von **Rekursion**.

*Complete the function compute\_pascal by using **recursion**.*



```
#include <iostream>
```

```
#include <cassert>
```

```
bool check_input_validity(int row, int pos){  
    if (row < pos || pos < 1){  
        std::cout << "Invalid: no element at the given row-position pair."  
        return false;  
    }  
    else {  
        return true;  
    }  
}
```

```
int compute_pascal(int row, int pos){  
    if (pos == 1){  
        return 1;  
    }  
    else if (pos == row){  
        return 1;  
    }  
    else {  
        return compute_pascal(row-1, pos) + compute_pascal(row-1, pos-1);  
    }  
}
```

```
void main(){
```

```
    //Input
```

```
    std::cout << "Please input a row and a position along the row: ";
```

```
    int row, pos;  
    std::cin >> row >> pos;
```

```
    //Check whether the input integers correspond to a valid entry
```

```
    assert (check_input_validity(row, pos));
```

```
    //Display the result
```

```
    std::cout << "The value at row " << row << " and position " << pos << " is "  
        << compute_pascal(row, pos);
```

```
}
```

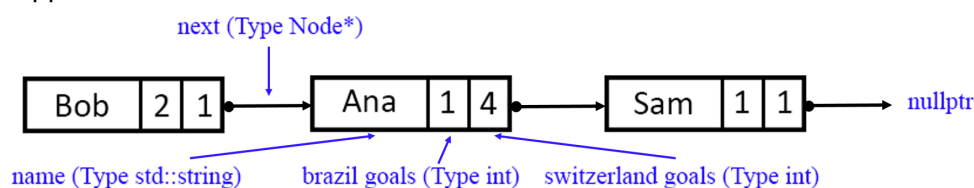


## Aufgabe 6: Datenstrukturen: WM Tipp-Spiel (15P)

Die Weltmeisterschaft ist vorbei und Sie und Ihre Freunde wollen die Wettresultate zum spannenden Spiel zwischen Brasilien und der Schweiz auswerten. Die abgegebenen Wetten sind in einer verketteten Liste abgelegt, implementiert mittels den untenstehenden `struct Node` und `struct List`. Ziel des folgenden Programmes ist es, die erste Person in der Liste zu finden, die das Resultat richtig getippt hat.

*The World Cup is over and you and your friends want to know who won the bet you did for that exciting Brazil vs Switzerland match. The bets have been stored in a linked list, implemented using `struct Node` and `struct List` below.*

*Goal of the following program is to find the first person in the list that bet the correct result.*



### Tasks

- /5P (a) Vervollständigen Sie die Funktion `add_friend`, welche Wetten zur Liste hinzufügt.
- /5P (b) Vervollständigen Sie die Funktion `find_winner`, welche durch die Liste geht und das passende Resultat sucht. Als Argumente wird das korrekte Resultat mitgeliefert.
- /5P (c) Vervollständigen Sie die Initialisierung der Liste in der Funktion `main` so, dass sie dem Bild oben entspricht und geben Sie die Programmausgabe an in der Box unter dem Code.

*Complete the function `add_friend` that appends a bet to the linked list.*

*Complete the function `find_winner` that goes through the list and searches for a matching result. The correct result is provided as arguments.*

*Complete the initialization of the list in the main function to match the image and give the output of the program in the box below the code.*

```
#include <iostream>
#include <string>
```

```
struct Node {
    std::string name;
    int brazil; int switzerland;
    Node* next;
    Node(std::string _name, int b, int s) : name(_name), brazil(b),
        switzerland(s), next(nullptr) {}
};
```

```
struct List{
    Node* first;
    Node* last;
    List() : first(nullptr), last(nullptr) {}
```

```

// PRE: Argument: a new bet (not null)
// POST: Adds the argument as last element of the list
void add_friend(Node* newBet){
    if( last == nullptr ){
        first = newBet;
    }else{
        last->next = newBet;
    }
    last = newBet;
}

// PRE: The correct score in b (for brasil) and s (for switzerland)
// POST: Returns the first node with the given scores, or nullptr otherwise
Node* find_winner(int b, int s) {
    Node* curr = first;
    while( curr != nullptr ){
        if( curr->brazil == b && curr->switzerland == s ){
            return curr;
        }
        curr = curr->next;
    }
    return nullptr;
}

};

int main () {
    List friends;
    friends.add_friend( new Node("Bob",2,1) );
    friends.add_friend( new Node("Ana",1,4) );
    friends.add_friend( new Node("Sam",1,1) );
    Node* winner = friends.find_winner(1,1);
    if(winner != nullptr){
        std::cout << winner->name << " won with match result: "
            << winner->brazil << "-" << winner->switzerland;
    }else{
        std::cout << "Nobody guessed it right!";
    }
    return 0;
}

```

Program output: Sam won with match result: 1-1

## Aufgabe 7: Objektorientierung: Reiche Studenten (13P)

Bearbeiten Sie die folgenden Teilaufgaben.

*Answer the following tasks.*

- /5P (a) Gegeben ist die folgende Klasse Student. Schreiben Sie eine Unterklasse EconomyStudent, welche die Methode checkFinances() überschreibt und die folgende Ausgabe auf der Konsole ausgibt: "I am making a lot of money."

*Given the following Student class, write a derived class called EconomyStudent, which uses the method checkFinances() and overrides it to print out the following text on the console: "I am making a lot of money."*

```
class Student {
public:
    virtual void checkFinances() {
        std::cout << "I have no money.\n";
    }
};
```

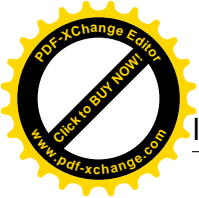
```
class EconomyStudent : public Student {
public:
    virtual void checkFinances() {
        std::cout << "I am making a lot of money.\n";
    }
};
```

- /4P (b) Kompiliert der untenstehende Code? Falls ja, was ist die Ausgabe auf der Konsole wenn er ausgeführt wird? Falls nein, warum nicht? Die Klassen Student und EconomyStudent sind wie in der Teilaufgabe a) definiert.

*Does the following code compile? If so, what is the output on the console when it is executed. If not, why not? The classes Student and EconomyStudent are defined according to the task a).*

```
EconomyStudent economyStudent;
economyStudent.checkFinances();
Student* student = &economyStudent;
student->checkFinances();
```

```
I am making a lot of money.
I am making a lot of money.
```



- (c) Kompiliert der untenstehende Code? Falls ja, was ist die Ausgabe auf der Konsole wenn er ausgeführt wird? Falls nein, warum nicht? Die Klassen Student und EconomyStudent sind wie in der Teilaufgabe a) definiert.

*Does the following code compile? If so, what is the output on the console when it is executed. If not, why not? The classes Student and EconomyStudent are defined according to the task a).*

/4P

```
Student student;  
student.checkFinances();  
EconomyStudent* economyStudent = &student;  
economyStudent->checkFinances();
```

Does not compile. Cannot initialize EconomyStudent \* with Student rvalue