

Name, Vorname:

Legi-Nummer:

Diese Selbsteinschätzung dient Ihrer und unserer Orientierung. Sie wird eingesammelt, korrigiert und vertraulich behandelt. Sie hat keinen Einfluss auf eine spätere Leistungsbewertung. **Sie haben 20 Minuten Zeit.**

Das folgende Kleingedruckte finden Sie auch auf einer "scharfen" Prüfung.

Allgemeine Richtlinien:

General guidelines:

- | | |
|--|--|
| <ol style="list-style-type: none"> 1. Dauer der Prüfung: 20 Minuten. 2. Erlaubte Unterlagen: Wörterbuch (für gesprochene Sprachen). Keine eigenen Notizblätter! Bei Bedarf stellen wir Ihnen weitere Blätter zur Verfügung. 3. Benützen Sie einen Kugelschreiber (blau oder schwarz) und keinen Bleistift. Bitte schreiben Sie leserlich. Nur lesbare Resultate werden bewertet. 4. Lösungen sind direkt auf das Aufgabenblatt in die dafür vorgesehenen Boxen zu schreiben (und direkt darunter, falls mehr Platz benötigt wird). Ungültige Lösungen sind deutlich durchzustreichen! Korrekturen bei Multiple-Choice Aufgaben bitte unmissverständlich anbringen! Lösungen auf Notizblättern werden nicht berücksichtigt. 5. Es gibt keine Negativpunkte für falsche Antworten. 6. Störungen durch irgendjemanden oder irgendetwas melden Sie bitte sofort der Aufsichtsperson. 7. Wir sammeln die Prüfung zum Schluss ein. Wichtig: stellen Sie unbedingt selbst sicher, dass Ihre Prüfung von einem Assistenten eingezogen wird. Stecken Sie keine Prüfung ein und lassen Sie Ihre Prüfung nicht einfach am Platz liegen. Dasselbe gilt, wenn Sie früher abgeben wollen: bitte melden Sie sich lautlos, und wir holen die Prüfung ab. Vorzeitige Abgaben sind nur bis 15 Minuten vor Prüfungsende möglich. 8. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen. 9. Wir beantworten keine inhaltlichen Fragen während der Prüfung. Kommentare zur Aufgabe schreiben Sie bitte auf das Aufgabenblatt. 10. Richtig beantwortete Multiple-Choice Fragen erhalten jeweils 1 Punkt. Es gibt keine negativen Punkte in MC Fragen. | <p><i>Exam duration: 20 minutes.</i></p> <p><i>Permitted examination aids: dictionary (for spoken languages). No sheets of your own! We will give you extra sheets on demand.</i></p> <p><i>Use a pen (black or blue), not a pencil. Please write legibly. We will only consider solutions that we can read.</i></p> <p><i>Solutions must be written directly onto the exam sheets in the provided boxes (and directly below, if more space is needed). Invalid solutions need to be crossed out clearly. Provide corrections to answers of multiple choice questions without any ambiguity! Solutions on extra sheets will not be considered.</i></p> <p><i>There are no negative points for false answers.</i></p> <p><i>If you feel disturbed by anyone or anything, let the supervisor of the exam know immediately.</i></p> <p><i>We collect the exams at the end. Important: you must ensure that your exam has been collected by an assistant. Do not take any exam with you and do not leave your exam behind on your desk. The same applies when you want to finish early: please contact us silently and we will collect the exam. Handing in your exam ahead of time is only possible until 15 minutes before the exam ends.</i></p> <p><i>If you need to go to the toilet, raise your hand and wait for a supervisor.</i></p> <p><i>We will not answer any content-related questions during the exam. Please write comments referring to the tasks on the exam sheets.</i></p> <p><i>Correctly answered Multiple-Choice questions receive 1 point each. There are no negative points in MC questions</i></p> |
|--|--|

Aufgabe	1	2	3	4	5	Σ
Punkte						
Maximum	4	6	5	6	8	29

1 Characters (4 Punkte)

Betrachten Sie die folgende Schleife und geben Sie ihre Ausgabe direkt in dem dafür vorgesehenen Kästchen an. Nehmen Sie an, dass Buchstaben gemäss dem ASCII-Code sortiert sind!

Consider the following loop and write its output directly in the provided box. Assume that letters are ordered according to the ASCII code!

```
for (char c = 'F'; c < 'R'; c += 3)
    std::cout << c;
```

Ausgabe *output*:

2 Referenzen (5 Punkte)

Kreuzen Sie an, ob die Aussagen richtig oder falsch sind.

Mark if the statements are correct or wrong.

Aussage <i>Statement</i>	correct richtig	wrong falsch
Referenzen können benutzt werden, um bei Funktionsaufrufen unnötige Kopien der Argumente zu verhindern. <i>References can be used to prevent unnecessary copies of the arguments of function calls.</i>		
Eine nicht const-Referenz kann nicht verwendet werden, um eine const-Variable zu referenzieren. <i>A non-const reference cannot be used to refer to a const variable.</i>		
Eine Referenz muss immer mit einem L-Wert initialisiert werden. <i>A reference must always be initialized with an L-value.</i>		
Eine Funktion sollte niemals eine Referenz auf eine lokale Variable zurückgeben. <i>A function should never return a reference to a local variable.</i>		
Nachdem eine Referenzvariable initialisiert wurde, kann sie jede andere Variable des gleichen Typs referenzieren. <i>Once a reference variable has been initialized, it can refer to any other variable of the same type.</i>		

3 EBNF I (5 Punkte)

Die folgende EBNF definiert eine Sprache zur Beschreibung von kantengewichteten Bäumen. Ein geklammerter Ausdruck steht für einen Baum, ganze Zahlen bezeichnen Kantengewichte. Ein illustrierendes Beispiel ist unter der EBNF zu finden.

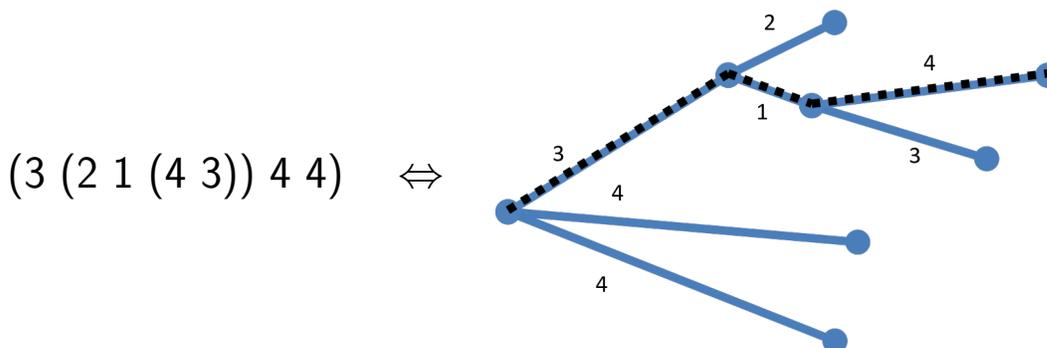
Beantworten Sie die Fragen auf der rechten Seite.

The EBNF displayed below defines a language for trees with weighted edges. An expression in parentheses stands for a tree, integer numbers designate edge weights. An illustrating example is displayed below the EBNF.

Answer the questions on the right side.

```
Tree = '(' Branches ')'.
Branches = Branch { Branch }.
Branch = Length | Length Tree.
Length = unsigned int.
```

Beispiel / *Example*:



Die gestrichelten Linien kennzeichnen Pfad mit maximaler Länge. Das ist nur für Aufgabe 4 relevant.

The dashed lines designate the path with maximal length. This is only relevant for problem 4.

- (a) Wahr oder falsch? *true or false?* 1 P

Die folgende Zeichenkette ist ein gültiger Baum (Tree) nach der EBNF:

The following string is a valid tree (Tree) according to the EBNF:

(10 20 30)

- (b) Wahr oder falsch? *true or false?* 1 P

Die folgende Zeichenkette ist ein gültiger Baum (Tree) nach der EBNF:

The following string is a valid tree (Tree) according to the EBNF:

()

- (c) Wahr oder falsch? *true or false?* 1 P

Die folgende Zeichenkette ist ein gültiger Baum (Tree) nach der EBNF:

The following string is a valid tree (Tree) according to the EBNF:

(10 5 (20 30 40) 30 40 (5 2 1) 45)

- (d) Wahr oder falsch? *true or false?* 1 P

Die folgende Zeichenkette ist ein gültiger Baum (Tree) nach der EBNF:

The following string is a valid tree (Tree) according to the EBNF:

10 (10 20 30)

- (e) Zeichnen Sie den Baum, welcher zu folgender Eingabe gehört. 1 P
Draw the tree that belongs to the following input.

(1 (2 (3) 4) 5)

4 EBNF II (6 Punkte)

Die folgende main-Funktion soll zu einem Baum, welcher gemäss der EBNF der vorigen Aufgabe am Eingabestrom vorliegt, die maximale Pfadlänge ausgeben. Die Länge eines Pfades ist die Summe der Gewichte entlang des Pfades. Ein Pfad eines Baumes geht jeweils von der Wurzel des Baumes bis zu einem Blatt. Im Beispiel auf Seite 4 ist der Pfad mit maximaler Länge $3+1+4=8$ gestrichelt eingezeichnet.

Vervollständigen Sie den Code der Funktionen Tree, Branches und Branch entsprechend (jeweils 2P).

The following main function shall return the maximal path length of a tree that is provided at the input stream according to the EBNF defined in the previous task. The length of a path is given by the sum of its weights. A path of a tree starts from the tree root and ends at a leaf. In the example on page 4 the path with maximal length $3+1+4=8$ is dashed.

Complement the code of functions Tree, Branches and Branch accordingly (2P each).

```
#include<iostream>
```

```
// code on the right hand side ---->
```

```
int main() {  
    int length = Tree(std::cin);  
    std::cout << "Maximal path length= " << length << "\n";  
    return 0;  
}
```

```

// POST: leading whitespace characters are extracted from is, and the
//      first non-whitespace character is returned (0 if there is none)
char lookahead (std::istream& is);

// Tree = '(' Branches ')'.
// Returns length of longest outgoing path.
int Tree (std::istream& is){
    char c;
    is >> c; assert(c=='(');
    int length = ;
    is >> c; assert(c==')');
    return length;
}

// Branches = Branch { Branch }.
int Branches (std::istream& is){
    int length = Branch(is);
    while (lookahead(is) != ')'){
        int bLength = Branch(is);
        
    }
    return length;
}

// Branch = Length | Length Tree.
int Branch(std::istream& is){
    int length = Length(is);
    if (lookahead(is) == '(') return ;
    return length;
}

// Length = unsigned int.
// Returns integer present at input stream
int Length (std::istream& is);

```

5 Vektoren und Referenzen (8 Punkte)

Seien x und y zwei Folgen mit Längen n und m , $x = (x_1, x_2, \dots, x_n)$ und $y = (y_1, y_2, \dots, y_m)$. Wir definieren den Reissverschluss von x und y als

$$\text{zip}(x, y) = (x_1, y_1, x_2, y_2, \dots, x_{\min(n,m)}, y_{\min(n,m)}).$$

Gesucht ist eine Funktion, die zwei Folgen ganzer Zahlen x und y als Zeigerbereiche erhält und $\text{zip}(x, y)$ ausgibt. Hier ist ein Beispiel:

$$x = (1, 3, 5, 7, 9, 11), y = (2, 4, 6, 8) \Rightarrow \text{zip}(x, y) = (1, 2, 3, 4, 5, 6, 7, 8).$$

Unten finden Sie das Skelett eines Programms, das eine solche Funktion definiert und aufruft. Ihre Aufgabe ist es, die Lücken so zu füllen, dass sich eine korrekte Funktion und ein korrektes Programm ergibt.

Let x and y be two sequences of lengths n and m , $x = (x_1, x_2, \dots, x_n)$ and $y = (y_1, y_2, \dots, y_m)$. We define the zip of x and y as

We are looking for a function that receives two integer sequences x and y as ranges and outputs $\text{zip}(x, y)$. Here is an example:

Above, you find a skeleton of a program that defines and calls such a function. Your task is to fill the gaps in such a way that you get a correct function and a correct program.

```
// POST: writes the zip of xVec and yVec
void zip (const std::vector<int>& xVec, const std::vector<int>& yVec)
{
    unsigned int xInd = ;
    unsigned int yInd = ;
    while (  ) {
        std::cout <<  << ' ' <<  << ' ';
        ++xInd;
        ++yInd;
    }
}

int main ()
{
    std::vector<int> x = {1, 3, 5, 7, 9, 11};
    std::vector<int> y = {2, 4, 6, 8};
    zip (  ); // 1 2 3 4 5 6 7 8
    return 0;
}
```