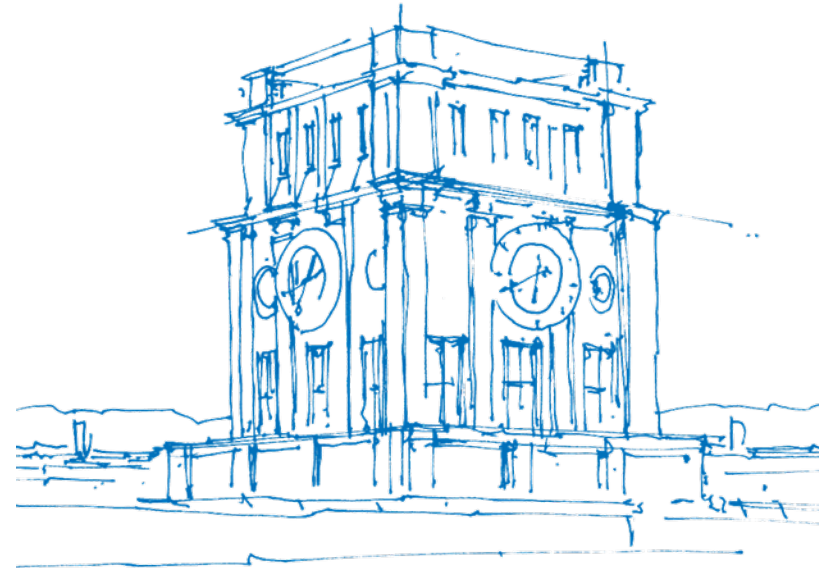


Praktikum: Grundlagen der Programmierung

Wenjie Hou

Technical University of Munich
Fakultät für Informatik

2. Tutorübung



TUM Uhrenturm

Overview

1. Lecture Review

2. P01: Summieren

2.1 Flow Diagram

2.2 Code - 1

2.3 Code - 2

3. P02: Integer factorization

3.1 Flow Diagram

3.2 Code

4. P03: Population of Rabbit

4.1 Code

5. P04: 3 and 7

5.1 Code

Lecture Review

Control Statements in Java

1 Decision Making

- if...then
- if...then...else
- if...then...else if...then...
- Switch
- Ternary operation $A ? B : C$

2 Looping Statements

- While Loop
- Do-While Loop
- For Loop
- Foreach Loop

3 Branching Statements

- Break
- Continue
- Return

P01: Summieren

In this task, a program is to be created which calculates the sum of several integers entered by the user. Ask the user to enter a number with "*Bitte Zahl eingeben :*" **until 0 is entered**.

As soon as the number 0 is entered, print "*Summe :*", print the sum of the numbers entered by the user in another line and **terminated** the program.

Example Output:

```
1 <Bitte Zahl eingeben :  
2 >-3  
3 <Bitte Zahl eingeben :  
4 >31  
5 <Bitte Zahl eingeben :  
6 >0  
7 <Summe :  
8 <28
```

Flow Diagram

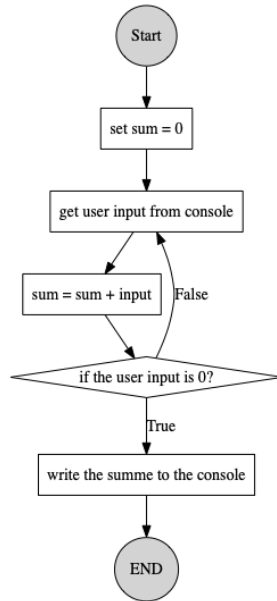


Figure: v1

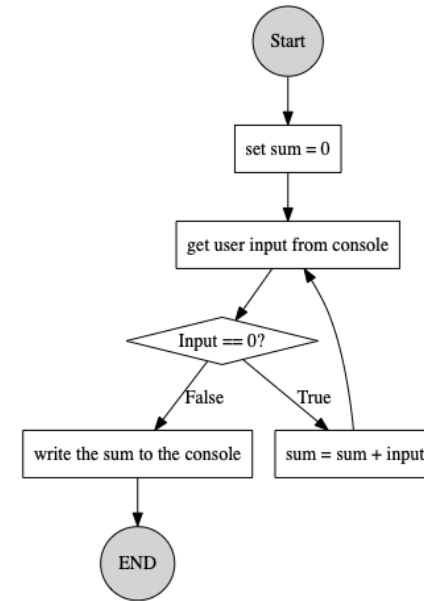


Figure: v2

Code -1



```
int sum = 0;

while (true){
    int input;
    input = MiniJava.readInt("Bitte Zahl eingeben:");
    sum += input;
    if (input == 0)
        break;
}
MiniJava.write("Summe:\n" + sum);
```

Code -2



```
int sum = 0;

int input = 0;
input = MiniJava.readInt("Bitte Zahl eingeben:");

while (input != 0){
    sum += input;
    input = MiniJava.readInt("Bitte Zahl eingeben:");
}

MiniJava.write("Summe:\n" + sum);
```

P02: Integer factorization

Any natural number $n > 1$ is either a prime itself or can be represented as a product of prime numbers. To calculate the prime factorization of a number n , divide it by all natural numbers starting from 2. If a divisor t is found, then output t . If the quotient $\frac{n}{t} > 1$, then it is also decomposed into its prime factors.

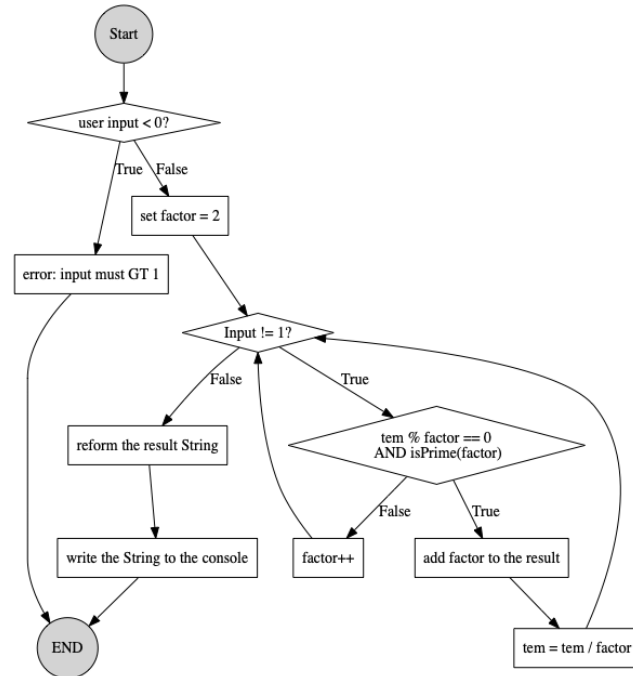
Write a MiniJava program that says *"Bitte Zahl eingeben:"* and reads a natural number $n > 1$ from the user and breaks it up into prime terms. Your program should output all prime factors of the number n separated by spaces on the screen. If the input number entered is $n \leq 1$, then print *"Fehler: $n > 1$ erwartet!"*.

Tip: You can use the `writeConsole()` method to print something on the console without extra line break. By using method `writeLineConsole()` you can print a line with the line feed.

Example Output:

```
1 Bitte Zahl eingeben:
2 >12321342
3 2 3 3 3 11 20743
```


Flow Diagram



Code

```
public static boolean isPrime(int num){
    for (int i = 2; i < num; i++){
        if (num % i == 0)
            return false;
    }
    return true;
}

public static void factorisation(int num){
    if (num < 1)
        MiniJava.write("Fehler: n>1 erwartet!");
    int factor = 2;
    int tem = num;
    ArrayList<Integer> results = new ArrayList<>();
    while (tem != 1){
        if (tem % factor == 0 && isPrime(factor)){
            results.add(factor);
            tem = tem / factor;
        } else
            factor++;
    }
    String str = results.stream().map(Object::toString).collect(Collectors.joining(" "));
    MiniJava.write(str);
}
```

P03: Population of Rabbit

We put a pair of mature rabbits on a desert island to find out how many rabbits will be born within a year. It is assumed that every mature couple will give birth to a new pair of rabbits each month. Each rabbit pair is sexually mature for the first month of life and each rabbit has **a lifetime of 3 months**.

Write a MiniJava program that reads in the number n , which is the n^{th} month. Your program should print out the sum of the rabbit pairs, which are mature in the n^{th} month. You can assume that $n \geq 1$.

Example Output:

```
1 <Bitte Zahl eingeben:
2 >3
3 <4
```

Code

```
public static void main(String[] args){
    int n = readInt("Geben Sie n ein:");

    int[] rabbits = new int[3];
    rabbits[0] = 1;

    int month = 1;

    while(month < n){
        int sum = 0;
        int i=0;
        while(i<3){
            sum += rabbits[i];
            i++;
        }

        rabbits[2] = rabbits[1];
        rabbits[1] = rabbits[0];
        rabbits[0] = sum;
        month++;
    }
    write(rabbits[0]+rabbits[1]+rabbits[2]);
}
```

P04: 3 and 7


Write a program that prints *"Bitte Zahl eingeben:"* and then read in a number n . Then it calculate the sum of all positive numbers less than or equal to n , and are divisible by 3 or 7. Then print the result to console. If the user enters a negative number, *"Fehler: $n > 1$ erwartet!"* and exit the program.

If the user types e.g. If the number is 25, so we add 3, 6, 7, 9, 12, 14, 15, 18, 21, and 24 and it gives us 129.

Example Output:

```
1 Bitte Zahl eingeben:
2 >91
3 1822
4
5 Bitte Zahl eingeben:
6 >16
7 1822
```

Code



```
public static void main(String[] args) {  
    int input = MiniJava.readInt();  
    int sum = 0;  
  
    for (int i = 0; i < input; i++) {  
        if (i % 3 == 0 || i % 7 == 0)  
            sum += i;  
    }  
    MiniJava.write(sum);  
}
```

Thank You!