University of Zurich[UZH]

s.e.a.l. software evolution & architecture lab

Informatics I – EProg HS15

Exercise 4

1 Task: Code Comprehension

1.1 Learning Objectives

1. understand loops and if-conditions
2. train code-reading and -comprehension

1.2 Assignment

Which of the following thirteen code snippets are syntactically correct and what is the output in those cases? Use the following additives:

- Walter Savitch, Java: An Introduction to Problem Solving & Programming
- Java API

Motivate your answers.

```
1 int sum = 0;
2 for(i=0;i<4;i++){
3 sum += i;
4 }
5 System.out.println(sum);
```

Listing 1: Snippet 1

Solution:

Wrong: i is not declared

```
1 int sum = 0;
2 for(int i=0;i<7;i++){
3 sum += i;
4 }
```

Listing 2: Snippet 2

Solution:

Correct: 21

```
1 int sum = 0;
2 for(int i=5;i>0;i--){
3 sum += i;
4 }
5 System.out.println(sum);
```

Listing 3: Snippet 3

Solution:

Correct: 15

```
1 for(int i=5;i>1;i--){
2 System.out.print(i);
3 }
```

Listing 4: Snippet 4

Solution:

Correct: 5432

```
1 int j = 0;
2 for(int i=0;i<5;j++){
3 System.out.print(j);
4 }
```

Listing 5: Snippet 5

Solution:

Syntactically correct, but it is an infinite loop ("0123456789101112...")

```
1 for(int i=0;i<10;i++){
2    i++;
3    System.out.print(i);
4 }
```

Listing 6: Snippet 6

Solution:

Correct: 13579

```
1 int sum = 0;
2 for(int i=0;i<7;i++)
3 sum++;
4 sum++;
5 System.out.println(sum);
```

Listing 7: Snippet 7

Solution:

Correct: 8 (curly brackets around first sum++ are omitted)

```
1 int sum = 0;
2 for(int i=0;i<7;i++)
3 sum += i;
4 sum += i;
5 System.out.println(sum);
```

Listing 8: Snippet 8

Solution:

Wrong: The second 'sum += i' is outside of the loop, but i is declared inside the loop and therefor not accessible outside.

```
1 int b = 10;
2 String s = if ( b % 2 == 0 ) {
3 "Even";
4 } else {
5 "Odd";
6 }
7 System.out.println(s);
```

Listing 9: Snippet 9

Solution:

Wrong: This syntax is not allowed in Java.

```
1 int a = 4;
2 if( a < 10 ){
3 a = 9;
4 } else if ( a < 5 ) {
5 a = 8;
6 } else {
7 a = 7;
8 }
9 System.out.println(a);
```

Listing 10: Snippet 10

Solution:

Correct: 9

```
1 Integer a = null;
2 if( (a = 3) > 0 ){
3 System.out.println(a);
4 }
```

Listing 11: Snippet 11

Solution:

Correct: 3 (But don't do this!!!)

```
1 int sum = 1;
2 int i;
3 for(i=0;(i<10 && sum < 100);i++){
4    sum *= 2;
5 }
6 System.out.println(i);
7 System.out.println(sum);
```

Listing 12: Snippet 12

Solution:

Correct: 7 and 128

```
1 int count = 0;
2 for(int i=0;i<5;i++)
3    for(int j=0;j<i;j++)
4       for(int k=0;k<j;k++)
5          count++;
6 System.out.println(count);
```

Listing 13: Snippet 13

Solution:

Correct: 10

2 Task: Primitive Datatypes, Binary Values and Logical Operators

2.1 Learning Objectives

1. understand how Java internally represents primitive datatypes
2. understand logical operators
3. train conversion to binary representation

2.2 Binary Code of Numbers

Java internally uses the two's complement to represent `byte`, `short`, `int` and `long`. What is the respective binary representation of the following `int` value?

```
1 int a = - 1555;
```

Listing 14: An integer value

Solution:

Binary code of 1555:	0000 0000 0000 0000 0000 0110 0001 0011
One's complement of 1555:	1111 1111 1111 1111 1111 1001 1110 1100
Two's complement of 1555:	1111 1111 1111 1111 1111 1001 1110 1101

When casting a bigger datatype to a smaller one, the supernumerary leftmost bits are stripped by Java.
Use this matter of fact as well as your knowledge about the size of Java data types and your knowledge from the lecture to explain the following casts in detail.

```
1 int a = 113857;
2 short b = (short) a;
3 System.out.println(b);
```

Listing 15: Casting int to short

Solution:

a = 113857 = 0000 0000 0000 0001 1011 1100 1100 0001
-> take the last 16 bits that fit into a short: 1011 1100 1100 0001
-> value of the first bit is one -> take Two's complement:
-> One's complement of above value: 0100 0011 0011 1110
-> Two's complement of above value: 0100 0011 0011 1111
-> (short) a = -17215

http://de.wikipedia.org/wiki/Integer_(Datentyp)

```
1 int a = 99857;
2 byte b = (byte) a;
3 System.out.println(b);
```

Listing 16: Casting int to byte

Solution:

a = 99857 = 0000 0000 0000 0001 1000 0110 0001 0001
-> take the last 8 bits that fit into a byte: 0001 0001
-> value of the first bit is zero:
(byte) a = 0001 0001 = 17

http://de.wikipedia.org/wiki/Integer_(Datentyp)

2.3 Binary Operations

The operator `&` conducts a bit wise `AND` conjunction of two numbers. The operator `|` conducts a bit wise `OR` disjunction of two numbers. This means, that every bit at position x of the left number is (AND respectively OR) concatenated with the bit at position x of the right number. The result is a sequence of bits that builds a new number. Think about the logical concatenations of boolean values that you already know. A bit value of `1` correlates to `true` and a bit value of `0` to `false`. The following Listing shows an example:

```
1  short a = 11;
2  short b = 13;
3  short c = a & b;
4
5  // Solution:
6  // a = 11 = 1011
7  // &
8  // b = 13 = 1101
9  // -------------
10 // c = 09 = 1001
```

Listing 17: Sample Binary Operation

Determine the missing variable in the code snippets below such that the bit-wise operation causes the given result. Example:

```
1 // Sample Exercise: c should be 1. What are possible values for b?
2
3 short a = 3;
4 short b = ???;
5 short c = a & b;
6
7 // Solution:
8 // a is 0000 0000 0000 0011 (3) and c should be 0000 0000 0000 0001 (1).
```

```
9  // Therefor b is 0000 0000 0000 0001 (1), because:
10 // a = 3 = 0000 0000 0000 0011
11 // &
12 // b = 1 = 0000 0000 0000 0001
13 // ------------------------------
14 // c = 1 = 0000 0000 0000 0001
```

Listing 18: Sample Binary Operation Exercise

Consider that there may be multiple solutions and try to find all of them.

```
1 // Exercise: c should be 12. What are possible values for b?
2 short a = 62;
3 short b = ???;
4 short c = a & b;
```

Listing 19: Binary Operation Exercise 1

Solution:

12

```
1 // Exercise: c should be 30. What are possible values for b?
2 short a = 8
3 short b = ???;
4 short c = a | b;
```

Listing 20: Binary Operation Exercise 2

Solution:

22, 30

```
1 // Exercise: d should be 15. What are possible values for b?
2 short a = 5;
3 short b = 2;
4 short c = ???;
5 short d = a | b | c;
```

Listing 21: Binary Operation Exercise 3

Solution:

8, 9, 10, 11, 12, 13, 14, 15

3 Task: Arrays

3.1 Learning Objectives

1. practice the creation and population of arrays in Java
2. practice looping over array elements

3.2 Arrays of primitve types

Write a small sample application with a `main` method that performs the following operations:

1. Creates an array that holds 10 integer values
2. Inserts the numbers 0 to 9 into the array
3. Calculates the sum of the values inside the array using a for loop

Solution:

```
1  public class PrimitiveArray {
2    public static void main(String[] args) {
3      int[] arr = new int[10];
4      for (int i = 0; i < arr.length; i++) {
5        arr[i] = i;
6      }
7      int sum = 0;
8      for (int n : arr) {
9        sum += n;
10     }
11     System.out.println(sum);
12   }
13 }
```

Listing 22: PrimitiveArray

Additional questions:

1. What happens if you accidentally loop beyond the last element of the array?
2. How can you ensure that you never loop beyond the last element?
3. Is it possible to enlarge an existing array?

Solution:

1. The JVM throws a ArrayIndexOutOfBoundsException at runtime!
2. Use for loops that reference array.length or use the for (int i : arr) notation
3. No, you have to create a new one and copy the elements. In the future, you will learn about ArrayList, which is extensible.

3.3 Storing books in a library

Write a class `Book` that has the String properties `title` and `author`. Then, write a class `Library` which represents a collection of books. The library class should use an array `private Book[] books = new Book[5];` to hold different books, starting with an array size of 5. The library class should have a method `addBook(Book book)` which is used to add books to the library. The method `addBook` should first check if the array is already full. If it is, it should create a new array, twice as big, copy the elements from the old array to the new array, and then replace the array in the library. This way, any number of books can be added. The library class should also offer a method for printing the authors and titles of all books contained within.

Write a TestDriver for your Library which adds 6 books and then prints out the authors and titles for all books.

Solution:

```
1  public class Book {
2    private String title;
3    private String author;
4
5    public void setTitle(String title) { this.title = title; }
6    public String getTitle() { return title; }
7
8    public void setAuthor(String author) { this.author = author; }
9    public String getAuthor() { return author; }
10 }
```

Listing 23: Book

```java
public class Library {
  private Book[] books = new Book[5];
  private int numberOfBooks = 0;

  private boolean isLibraryFull() {
    return books.length <= numberOfBooks;
  }

  public void addBook(Book book) {
    if (isLibraryFull()) {
      Book[] copy = new Book[books.length*2];
      for (int i = 0; i<books.length; i++) {
        copy[i] = books[i];
      }
      books = copy;
    }
    books[numberOfBooks++] = book;
  }

  public void printLibrary() {
    for (Book b : books) {
      if (b != null) {
        System.out.println(b.getAuthor() + ": " + b.getTitle());
      }
    }
  }
}
```

Listing 24: Library

```java
public class TestLibrary {
  public static void main(String[] args) {
    Library lib = new Library();
    for (int i = 1; i < 7; i++) {
      Book book = new Book();
      book.setAuthor("author" + i);
      book.setTitle("title" + i);
      lib.addBook(book);
    }
    lib.printLibrary();
  }
}
```

Listing 25: TestLibrary

4 Task: Programming

4.1 Learning Objectives

1. Combine the knowledge about object oriented programming and technical basics and apply it.
2. Get to know methods from `String`, `Math` and `Integer`
3. Work with loops and if-conditions

4.2 BinaryConverter

Write a class `BinaryConverter` that is able to convert a given decimal number into a binary number. The given decimal number should be converted into a standard binary if the decimal value is positive and into Two's complement if the decimal value is negative. The result shall be printed to the console. Use the following class scaffolds:

```
1 class BinaryConverter {
2     ...
3     /*
4      * int dNumber: The decimal number
5      */
6     public String convert(int dNumber)
7     ...
8 }
```

Remarks

1. Reflect how you would do the conversion handwritten and try to implement this process.
2. You should use methods of the Java class `Math` which provides various mathematical operations. Find information about that in Java Doc.
3. You should use methods of the Java class `String`. Consider a String as a sequence of `char`'s. The class `String` allows you to access the individual characters. Find the proper methods in the Java Doc of the `String` class.
4. You can use the method `valueOf` of the class `Integer` to convert a string that represents a decimal value into an `Integer`. Example: `Integer a = Integer.valueOf("42").`
5. You can use the method `getNumericValue` of the class `Character` to convert a `char` into its numerical value. Find information in the Java Doc of this method.

4.3 TestDriver

Write a class `BinaryConverterTestDriver` and test the class `BinaryConverter` with different values.

Solution:

```java
public class BinaryConverterTestDriver {

  public static void main(String[] args) {

    BinaryConverter binaryConverter = new BinaryConverter();

    int dNumber = 4;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));

    dNumber = -4;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));

    dNumber = 352;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));

    dNumber = -352;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));

    dNumber = 88;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));

    dNumber = -88;
    System.out.println("Test 1 : dNumber -> " + dNumber + " -> binary
        number: " + binaryConverter.convert(dNumber));
  }
}
```

Listing 26: BinaryConverterTestDriver

```java
public class BinaryConverter {

   public String convert(int dNumber) {

      // Positive or negative?
      boolean positive = (dNumber >= 0);

      String result;

      if (positive) {
         result = Integer.toBinaryString(dNumber);
      }
      else{
         dNumber = 0 - dNumber;
         String onesComplement = Integer.toBinaryString(~dNumber);
         String twosComplement = "";
         boolean carry = true;
         for (int i = onesComplement.length() - 1; i >= 0; i--) {
            if (carry) {
               if (Character.getNumericValue(onesComplement.charAt(i))
                   == 0) {
                  twosComplement = "1" + twosComplement;
                  carry = false;
               }
               else{
                  twosComplement = "0" + twosComplement;
               }
            }
            else{
               twosComplement = onesComplement.charAt(i) +
                   twosComplement;
            }
         }
         result = twosComplement;
      }
      return result;
    }
}
```

Listing 27: BinaryConverter