FREIE UNIVERSITÄT BOZEN
LIBERA UNIVERSITÀ DI BOLZANO
FREE UNIVERSITY OF BOZEN · BOLZANO

Fakultät für Informatik | Facoltà di Scienze e tecnologie informatiche | Faculty of Computer Science

Introduction to Programming

Written Examination

23.9.2016

FIRST NAME		**LAST NAME**	
STUDENT NUMBER		**SIGNATURE**	

Instructions for students:

Write First Name, Last Name, Student Number and Signature where indicated. If not, the examination can not be marked.

Do not speak to any other student during the examination. If you speak to another student, your examination will be cancelled.

Use a pen, not a pencil.

Write neatly and clearly.

Reply to the following questions. You cannot consult any material.

1. When **executing** a program, the processor reads each program instruction from

A) secondary memory (storage)
B) the Internet
C) registers stored in the processor
D) main memory
E) could be any of these

Answer: D
Explanation: D) The program is first loaded from secondary memory into main memory before it is executed so that the processor is not slowed down by reading each instruction. This idea of executing programs stored in memory is called the Stored Program Computer and was pioneered by John Von Neumann in the 1940s.

2. If in the following statements the casting to `float` is **removed** would this raise an exception? Explain what would happen with and without the casting and what would be printed by the third statement in the two cases:

```
int total = 45;
float result = (float) total / 7;
System.out.println (result);
```

Res: No exception, result would be assigned to 6.0 (without cast) and 6.428 (with cast).

3. Write the output of these statements:

```
int base = 1;
int count = 3;
if (count < 4 || ++base < count)
   System.out.println(count);
System.out.println(base);
```

Res:
3
1

4. Write the output of these statements:

```
int a = 2;
System.out.println(++a + Math.pow(a, 2));
```

```
a = 2;
System.out.println(Math.pow(a, 2) + ++a);
```

Res:
12.0
7.0

5. Is the following code syntactically correct? If yes, write the output of these statements:

```
Integer num10 = 4;
Integer num20 = num10;
num10 = new Integer(5);
System.out.println(num20.intValue());
```

Res: 4

}

6. Assuming that c1 and c2 are Boolean variables, create a truth table for the expression:

(!c1 || c2) && (c1 || !c2)

Res: see the lecture slides.

7. Which of the following statements is completely true?
A) If a class is declared to be abstract then every method in the class is abstract and must be overridden
B) If a class is declared to be abstract then some methods in the class may have their bodies omitted
C) If a class is declared to be abstract then all methods in the class must have their bodies omitted
D) If a class is declared to be abstract then all the instance variables must be overridden when a concrete class is derived from the abstract base class

Answer: B
Explanation: B) The only way to create an abstract class is to create some abstract method within the class. So, answer B is true. Certainly all the methods in an abstract class don't have to be abstract.

8. What is printing the following code?

```
public static void main (String[] args)
{
      int lDigit, number = 7689, res=0;

      do
```

```
        {
              lDigit = number % 10;
              res = (res * 10) + lDigit;
              number = number / 10;
        }
        while (number > 0);

        System.out.println (res);
}
```

Res: is the reverse 9867

9. What is printing the following code?

```
for(int i = 9; i > 0; i--){
      for(int j = i; j < 9; j++){
          System.out.print("_");
      }
      for(int j = i; j > 0; j--){
          System.out.print(i);
      }
      System.out.println();
  }
```

Res:
999999999
_88888888
__7777777
___666666
____55555
_____4444
______333
_______22
________1

10. What is printing the following code?

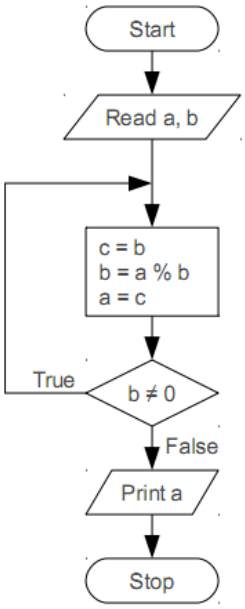
```
public class ParameterTester {

    public static void main(String[] args) {
        ParameterModifier modifier = new ParameterModifier();
        String a1 = "111";
        Num a2 = new Num(222);
        Num a3 = new Num(333);
        int a4 = 444;

        modifier.changeValues(a1, a2, a3, a4);

        System.out.println("a1\ta2\ta3\ta4");
        System.out.println(a1 +"\t" + a2 +"\t" + a3 +"\t" + a4
+"\n");
    }
```

```
    private static class ParameterModifier {
        public void changeValues(String f1, Num f2, Num f3, int f4) {
            f1 = "555";
            f2.setValue(888);
            f3 = new Num(777);
            f4 = 999;
        }
    }

    private static class Num {
        private int value;

        public Num(int update) {
            value = update;
        }

        public void setValue(int update) {
            value = update;
        }

        public String toString() {
            return value + "";
        }
    }
}
```

Res:

a1	a2	a3	a4
111	888	333	444

11. What is printing the algorithm described in the flowchart below when **a=8, b=3** and when **a=12, b=10**?



Res:
It computes the GCD. It prints 1 in the first case and 2 in the second.

12. Write a convenient equals method (overriding the equals method defined in the class Object) for the class declared below:

```
public class Bicycle {
      private int cadence;
      private int gear;
      private int speed;
}
```

Res:

```
public boolean equals(Object bicycle) {
  return  cadence == ((Bicycle)bicycle).cadence &&
          gear == ((Bicycle)bicycle).gear &&
          speed == ((Bicycle)bicycle).speed;
}
```

13. *An int array stores the following values. Use the array to answer these next questions.*

9	4	12	2	6	8	18

Show the array state after the **third** pass of the Insertion Sort algorithm? A "pass" is defined as one execution of the code in the **for** loop.

```
//-----------------------------------------------------------------
//  Sorts the specified array of objects using the insertion
//  sort algorithm.
//-----------------------------------------------------------------
   public static void insertionSort (Comparable[] list)
   {
      for (int index = 1; index < list.length; index++)
      {
         Comparable key = list[index];
         int position = index;

         while (position > 0 && key.compareTo(list[position-1]) < 0)
         {
            list[position] = list[position-1];
            position--;
         }

         list[position] = key;
      }
   }
```

Res: 2, 4, 9, 12, 6, 8, 18

14. Given the following method:

```
public void differentArray(float[] x)
  {
       x = new float[99];
       x[0] = 26.9f;
  }
```

what is the output after the following code is run?

```
float[] xx = new float[99];
xx[0] = 0.0f;
differentArray(xx);
System.out.println("xx[0] = " + xx[0]);
```

Res: 0.0

15. What is printed?

```
public class Inherit
{
      abstract class Figure
      {
            void display( )
```

```
        {
            System.out.println("Figure");
        }
    }

    abstract class Rectangle extends Figure
    {
    }

    class Box extends Rectangle
    {
        void display( )
        {
            System.out.println("BOX");
        }
    }

    Inherit( )
    {
        Figure f = new Box( );
        f.display( );
        Rectangle r = (Rectangle) f;
        r.display( );
    }

    public static void main(String[ ] args)
    {
        new Inherit( );
    }
}
```

Res:
BOX
BOX
"f" is a polymorphic reference variable, as is "r." Trace through the inheritance structure to see that both references' invocation of the display( ) method will, indeed, display "Rectangle."

16. 1) Interface classes cannot be extended but classes that implement interfaces can be extended. True or False? Explain.

Answer: FALSE
Explanation: Any class can be extended whether it is an interface, implements an interface, or neither. The only exception to this is if the class is explicitly modified with the word "final" in which case it cannot be extended.

17. Write a recursive method that prints the numbers 1…n (where n is the int parameter of the method) in ascending order:

```
public void asc(int n) {

    ...

}
```

```
Result:
public static void asc(int n) {
        if (n > 1) {
            asc(n - 1);
        }
        System.out.println(n);
    }
```

18. Write a recursive function to perform exponentiation return x^m, assuming m >= 0

```
 public int exp(int x, int m) {
}

Res:

public int exp(int x, int m) {
    if (m == 0) { return 1; }
    if (m == 1) { return x; }
    return x * exp(x, m-1);
}
```