UNIVERSITY OF COLOMBO, SRI LANKA

UCSC

UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING

BIT

DEGREE OF BACHELOR OF INFORMATION TECHNOLOGY

Academic Year 2015/2016 – 1st Year Examination – Semester 2

IT2205 - Programming I

22nd October, 2016

(TWO HOURS)

Important Instructions :

- The duration of the paper is **2 (two) hours**.
- The medium of instruction and questions is English.
- The paper has **45 questions** and **13 pages**.
- All questions are of the MCQ (Multiple Choice Questions) type.
- All questions should be answered.
- Each question will have 5 (five) choices with **one or more** correct answers.
- All questions will carry equal marks.
- There will be a penalty for incorrect responses to discourage guessing.
- The mark given for a question will vary from 0 (*All the incorrect choices are marked & no correct choices are marked*) to +1 (*All the correct choices are marked & no incorrect choices are marked*).
- Answers should be marked on the special answer sheet provided.
- Note that questions appear on both sides of the paper.
  If a page is not printed, please inform the supervisor immediately.
- Mark the correct choices on the question paper first and then transfer them to the given answer sheet which will be machine marked. **Please completely read and follow the instructions given on the other side of the answer sheet before you shade your correct choices.**

1) Select from among the following, the first name given for Java.

(a) **Oak**	(b) Odd	(c) Oslo
(d) J	(e) Kawa	

2) Select from among the following, valid features which are supported by Java.

(a) Exception Handling	**(b) Multithreading**
(c) Network Programming	(d) Operator Overloading
(e) Memory management	

3) Select from among the following, what JDK stands for.

(a) Java Design Kit	(b) Java Data Kit
(c) Java Definition Kit	**(d) Java Development Kit**
(e) Java Deployment Kit	

4) Select from among the following, valid output generated after a Java source file is successfully compiled.

(a) Object Code	**(b) Byte Code**
(c) P Code	(d) Java Compiler
(e) Java Interpreter	

5) Java is an Object Oriented Programming language. Select from among the following correct option(s) which show(s) how Java achieve common information sharing.

(a) Abstraction	(b) Pointer manipulation
(c) Exception Handling	**(d) Inheritance**
(e) Polymorphism	

6) Select from among the following, valid reference data types available in Java.

(a) **System**	(b) private	(c) **String**
(d) public	(e) **Scanner**	

Consider the following program written in Java to answer question 7 – 11.

```
public class Ex2{
public static void main(String args[]){

float value1=10.0f,value2=20.0f;
int valueOne = 30,valueTwo=40;

//System.out.print(value1 & value2);
System.out.println(valueOne ^ valueTwo);

  }
}
```

7) Select from among the following, correct option(s) which illustrate/s output of the program when it is executed.

(a) 54	(b) 55	(c) 56
(d) true	(e) error	

8) Assume that the following statement is written in the program removing the symbols // in the program

```
System.out.print(value1 & value2);
```

What would the output of the program be?

(a) 0 54	(b) 0	(c) 54
(d) 0.0 54	**(e) error**	

9) Select from among the following, valid symbol(s) available in Java programming language which has the similar functionalities as the symbol // illustrated in the above program.

(a) /* */	(b) \\	(c) \n
(d) /* **/	**(e) /** */**	

10) Select from among the following, names which are given to each operator used in the program.

(a) bitwise AND → &	**(b) exclusive OR →** ^	(c) comparison → =
(d) logical AND → &	(e) arithmetic plus → ^	

11) Select from among the following, valid option(s) which shows the same operator(s) belong to the same category as the operators & and ^.

(a) >>	**(b)** <<	**(c)** ~
(d) |	(e) <=	

Use the following declarations and initializations to evaluate the Java expressions given in questions 12 - 17. Assume that each expression is evaluated separately in the program.

```
float x = 10.0f;
int num1=20,num2=30;
double num4 = 100;
boolean value5 = true;
char ch = 'B'; // note that the ASCII value of A is 65
```

Select from among the given options, the correct output for each of the questions 12 –19.

12)
```
System.out.println("Answer"+value5+num1);
```

(a) true	(b) Answertrue	(c) 320
(d) Answertrue20	(e) error	

13)
```
System.out.println(value5+ch);
```

(a) trueB	(b) true66	(c) 82
(d) 63	(e) **error**	

14) `System.out.println(x*ch);`

(a) 660.0	(b) true	(c) 65
(d) 65.0	(e) error	

15) `System.out.println(num4 + num1 > ch);`

(a) 66	**(b) true**	(c) false
(d) 10020	(e) error	

16) `System.out.println(num4 > num1 > value5);`

(a) 12	(b) true	(c) 120
(d) false	**(e) error**	

17) `System.out.println(ch++ + ++ch + ch++);`

(a) 198	(b) 200	(c) 201
(d) 202	(e) error	

18) `System.out.println(num2>num1 && num4/2 >ch);`

(a) true	(b) 100	**(c) false**
(d) 30	(e) error	

19) System.out.println(x + ch);

(a) true	**(b) 76.0**	(c) 10.066
(d) 76	(e) error	

20) Select from among the following, correct operator(s) which use(s) to allocate memory to an array variable.

(a) malloc	(b) alloc	**(c) new**
(d) Object	(e) calloc	

21) Consider the following program written in Java.

```
public class Ex5{
public static void main(String args[]){

    char ar [] = new char[10];
      for (int i = 0; i < 5; ++i) {
           ar[i] = 'i';
                System.out.print(ar[i] + "");
    }
 }
}
```

What would the output of the program be?

(a) 12345	(b) 01234	**(c) iiiii**
(d) 105106107108109	(e) 4849505152	

22) Consider the following illustration.

```
* * * *
* * * *
* * * *
* * * *
```

Select from among the following, the valid program segment, by which the above illustration can be derived as an output in the command line.

```
(a)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         System.out.print("* ");
         System.out.println();}

(b)     for(int i=0;i<=3;i++){
          for(int k=0;k<=3;k++)
            System.out.println("* ");
        System.out.println();    }

(c)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         if(i<=k)
        System.out.print("*");
         else
        System.out.print(" ");
        System.out.println();
      }

(d)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
        if(i>=k)
             System.out.print("*");
        else
             System.out.print(" ");
             System.out.println();   }

(e)      for(int i=0;i<=3;i++){
        for(int k=0;k<=3;k++)
        if(i==k)
            System.out.print("*");
        else
            System.out.print(" ");
            System.out.println();    }
```

23) Consider the following illustration.

```
* * * *
* * *
* *
*
```

Select from among the following, the valid program segment, by which the above illustration can be derived as an output in the command line.

```
(a)    for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         System.out.print("* ");
         System.out.println();}

(b)    for(int i=0;i<=3;i++){
         for(int k=i;k<=3;k++)
           System.out.print("* ");
       System.out.println();   }

(c)    for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         if(i<=k)
        System.out.print("*");
         else
        System.out.print(" ");
        System.out.println();
      }

(d)    for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
        if(i>=k)
             System.out.print("*");
        else
             System.out.print(" ");
             System.out.println();  }

(e)     for(int i=0;i<=3;i++){
        for(int k=0;k<=3;k++)
        if(i==k)
            System.out.print("*");
        else
            System.out.print(" ");
            System.out.println();   }
```

24) Consider the following illustration.

```
*
 *
  *
   *
```

Select from among the following, the valid program segment, by which the above illustration can be derived as an output in the command line.

```
(a)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         System.out.print("* ");
         System.out.println();}

(b)     for(int i=0;i<=3;i++){
         for(int k=i;k<=3;k++)
           System.out.print("* ");
       System.out.println();   }

(c)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         if(i<=k)
        System.out.print("*");
         else
        System.out.print(" ");
        System.out.println();
      }

(d)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
        if(i>=k)
             System.out.print("*");
        else
             System.out.print(" ");
             System.out.println();  }

(e)      for(int i=0;i<=3;i++){
        for(int k=0;k<=3;k++)
        if(i==k)
            System.out.print("*");
        else
            System.out.print(" ");
            System.out.println();   }
```

25) Consider the following illustration.

```
*
**
***
****
```

Select from among the following, the valid program segment, by which the above illustration can be derived as an output in the command line.

```
(a)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         System.out.print("* ");
         System.out.println();}

(b)     for(int i=0;i<=3;i++){
         for(int k=i;k<=3;k++)
           System.out.print("* ");
       System.out.println();   }

(c)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         if(i<=k)
        System.out.print("*");
         else
        System.out.print(" ");
        System.out.println();
      }

(d)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
        if(i>=k)
             System.out.print("*");
        else
             System.out.print(" ");
             System.out.println();  }

(e)      for(int i=0;i<=3;i++){
        for(int k=0;k<=3;k++)
        if(i==k)
            System.out.print("*");
        else
            System.out.print(" ");
            System.out.println();   }
```

26) Consider the following illustration.

```
****
 ***
  **
   *
```

Select from among the following, the valid program segment, by which the above illustration can be derived as an output in the command line.

```
(a)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         System.out.print("* ");
         System.out.println();}

(b)     for(int i=0;i<=3;i++){
         for(int k=i;k<=3;k++)
           System.out.print("* ");
        System.out.println();   }

(c)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
         if(i<=k)
        System.out.print("*");
         else
        System.out.print(" ");
        System.out.println();
      }

(d)     for(int i=0;i<=3;i++){
       for(int k=0;k<=3;k++)
        if(i>=k)
             System.out.print("*");
        else
             System.out.print(" ");
             System.out.println();  }

(e)      for(int i=0;i<=3;i++){
        for(int k=0;k<=3;k++)
        if(i==k)
            System.out.print("*");
        else
            System.out.print(" ");
            System.out.println();   }
```

27) Select from among the following, correct statements on access specifies available in Java.

(a) public members of class can be accessed by any code in the program

(b) private members of class can only be accessed by other members of the class

(c) private members of class can be inherited by a sub class, and become protected members in sub class

(d) protected members of a class can be inherited by a sub class, and become private members of the sub class

(e) default members are defined by using the default key word in Java

Consider the following program written in Java to answer question 28 – 29.

```
public class Ex7{
public static void main(String args[]){
int ar1[]={1,2,3,4,5,6,7,8};
int ar2[]=new int[4];
int ar3[]=new int[4];
  for(int i=0,k=0;i<=7;i++)
     if(i<=3)
        ar2[i]=ar1[i];
     else{
        ar3[k]=ar1[i];
           k++;
         }
  for(int i=0;i<=3;i++){
      System.out.print(ar2[i]);
      System.out.print(ar3[i]);
  }
 }
}
```

28) Select from among the following, how many array declarations are written in the program.

(a) 1 (b) 2 (c) 3
(d) 4 (e) 5

29) What would the output of the program be?

(a) 12345678 (b) 87654321 (c) error
(d) 1234 **(e) 15263748**

Consider following program written in Java to answer question 30 – 35.

```
class A {
        private int i;
        public void display() {
            System.out.println(i);
        }
    }
    class B extends A {
        private int j;
        public void display() {
            System.out.println(j);
        }
    }
    class DP {
        public static void main(String args[]){
            B obj = new B();
            obj.i=1;
            obj.j=2;
            obj.display();
        }
   }
```

30) Select from among the following, valid description(s) which can be used to name the class denoted as **B** in the program.

(a) Super class	**(b) Child class**	**(c) Derived class**
(d) Sub class	(e) Base class	

31) Select from among the following, valid description(s) which can be used to name the class denoted as **A** in the program.

(a) Super class	(b) Child class	**(c) Parent class**
(d) Sub class	**(e) Base class**	

32) When the program was compiled, there were errors generated. Select from among the following, the statement(s) which caused to generate those errors.

(a) private int i;	**(b) private int j;**	`(c) B obj = new B();`
(d) class DP	(e) B obj = new B();	

33) When the program is interpreted, the programmer needs to get an output in the command prompt. Then the program needs modification. Select from among the following, valid modifications that can be applied to the program.

(a) int i;	**(b) int j;**	**(c) introduce getters**
(d) introduce setters	(e) remove main()	

34) Consider the following segment of code which is appearing in the program.

```
class B extends A
```

Select from among the following, valid option(s) which shows the object oriented feature which has been achieved by the given segment of code.

(a) encapsulation	(b) information hiding	(c) abstraction
(d) inheritance	(e) polymorphism	

35) In the program one can see the following segment of program in two places.

```
public void display() {  }
```

Select from among the following, the object oriented feature(s) which has been achieved through the program segment.

(a) encapsulation	**(b) overriding**	(c) overloading
(d) inheritance	**(e) polymorphism**	

36) Select from among the following, valid option which shows the super class of all the exceptional types.

(a) String	(b) System	**(c) throwable**
(d) catchable	(e) RuntimeException	

37) Select from among the following, valid option(s) which can be used to manually throw an exception.

(a) try	(b) finally	**(c) throw**
(d) catch	(e) main	

38) Select from among the following, the exception which occur when JVM runs out of memory.

(a) MemoryBoundException	(b) OutOfRangeError	(c) NullReferenceError
(d) OutOfMemoryError	(e) throw	

39) Consider the following program written in Java.

```
public class Ex12 {
  public static void main(String args[]){
    String name1 = "Manuja Gunasena";
    System.out.print(name1.length());
    String name2 = "Binusha Mallikarachchi";
    System.out.print(name2.charAt(5));
            }
        }
```

What would the output of the program be?

(a) 14h	**(b) 15h**	(c) 15s
(d) 6s	(e) 6h	

40) Select from among the following, valid access modifiers allowed in Java interfaces.

(a) static	**(b) public**	(c) private
(d) default	(e) protected	

41) Consider the following program written in Java.

```
public class Ex13 {
   public static void main(String args[]){
        String a = "hello i love java";
   System.out.println(a.indexOf('o')+" "+a.lastIndexOf('o'));
            }
        }
```

What would the output of the program be?

(a) 1 1	(b) 4 4	(c) 5 5
(d) 4 9	(e) 5 10	

42) Select from among the following, (the) class/es which used for input and output operations when work with bytes.

(a) **InputStream**	(b) Reader	(c) Read()
(d) FileWriter	(e) Writer	

43) Select from among the following, valid annotations used in Java.

(a) **@Override**	(b) public	(c) private
(d) protected	(e) **@Deprecated**	

44) Select from among the following, valid escape sequences used in Java.

(a) **\n**	(b) //	(c) \\
(d) **\b**	(e) /t	

45) Consider the following program written in Java.

```
class Ex45{
 public static void main(String args[]){
                 int x = 2;
                 int y = 0;
                 for ( ; y < 10; ++y) {
                     if (y % x == 0)
                         continue;
                     else if (y == 8)
                          break;
                     else
                 System.out.print(y + " ");
                 }
            }
}
```

What would the output of the program be?

(a) 8	(b) **1 3 5 7 9**	(c) 0 0 0 0 0 0
(d) 2 4 6 8 10	(e) error	

****