香 港 中 文 大 學
The Chinese University of Hong Kong

Course Examination 2nd term, 2011-12

Course Code & Title : CSCI1530 Computer Principles and Java Programming

Time allowed : 2 hours

Student I.D. No. : .... Seat No.: ....

Answer **ALL** Questions. Full Score is 100%. [*] indicates potential time-consuming problem.
Please mark all your answer on the space provided in this question-answer book.
You are recommended to write your student I.D. on each page.
List of operators and part of the Unicode table (ASCII table) are given in the last page.
Use the symbol ␣ to denote space in your answer where needed.

1. Complete the following method for determining if the parameter contains the following substring "123". If yes, return true; otherwise, return false. If the parameter is null or is an empty String, also return true. (10%)

```
public Boolean contains123( String s )
{







}
```

Marker's Use Only

Question	1	2	3	4	5	6	Total
Score							
Full	10	15	20	18	17	20	100

2. Answer the following questions.

a) What are the differences in syntax form and in function between a constructor and an ordinary method? (4%)

b) What is the compilation error for the following coding ? (3%)

```
class ExamTime{
       int v = 30;
       Public static void main( String[] args ){
           System.out.println(“value is ” + v);
       }
}
```

c) Let SpaceCar be a class that has been properly defined. (3 %)

```
SpaceCar  car1  car2;

car 1 = new SpaceCar();

car 2 = car1;
```

How many SpaceCar objects are created in the above ?

Answer:

```
SpaceCar  car1  car2;

car 1 = new SpaceCar();

car 1 = new SpaceCar();

car 2 = new SpaceCar();
```

How many SpaceCar objects are created in the above ?

Answer:

3) What will be printed by each of the following code segments ? (10 %)

```
public class exam{
    public static void main(String args[]){
    String s1 = “ABC”;
    String s2 = new String(“ABC”);
    if(s1 == s2)
       System.out.println(1);
    else
       System.out.println(2);
    }
}
```

Answer:

```
public class exam{
    public static void main(String args[]){
    String s1 = “ABC”;
    String s2 = “ABC”;
    if(s1 == s2)
       System.out.println(1);
    else
       System.out.println(2);
    }
}
```

Answer:

```
public class exam{
    public static void main(String args[]){
    int i = 3/2;
    switch(i){
         case 0: System.out.println(“aaa”); break;
         case 1: System.out.println(“bbb”);
         case 2: System.out.println(“ccc”); break;
         }
     }
}
```

Answer:

```
public class exam{
    public static void main(String args[]){
    int i = 3%2;
    do{
        i--;
    } while (i > 2);
    System.out.println(i);
    }
}
```

Answer:

```
public class exam{
    public static void main(String args[]){
    int i;
    double d = 3.7;
    i = ((int)d ) * ((int)Math.round(d));
    System.out.println(i);
    }
}
```

Answer:

```
int count = 1;
while ( count != 10 ) {
      count = count + 2;
}
System.out.println(count);
```

Answer:

3. What is the output of the following ? (5%)

```
public class Stars {
      Public static void main (String[] args)
      {
           int max = 5;
           for (int row =1; row <= max; row++)
           {
                    for (int star = 1; star <= row; star++)
                       System.out.print(“*”);
                    System.out.println();
           }
      }
}
```

Answer:

4. Rewrite the following nested if statement into aı equivalent switch statement. (5%)

```
if (number == 5)
      myChar = 'A';
else
      if (number == 6)
          myChar = 'B';
      else
          myChar = 'C';
```

Answer:

5. Complete the following code fragment so that it will print the prime numbers in array pnumbers in reverse order. You should use the array attribute "length". (10%)

```
public class Foo
{
   public static void main (String[] args)
   {
      int[] pnumbers = {2, 3, 5, 7, 11, 13, 17, 19};
```

6. Write a method that computes the following for integers a and n. (10%)

$$a^{n} + a^{n-1} + \ldots + a^{2} + a$$

The method accepts integers a and n as parameters and returns the result as type `long`, assume the result will not exceed the maximum value of a `long` type.

7. Finish the following method for printing a giant pattern in the following format based on the instance field integer **N** which is assumed to have been initialized. The method prints nothing and returns immediately if N is negative. You may NOT hard-code the pattern. (20%)

Four sample runs on different values of N:

```
M    |\/|   |\  /|  |\        /|
            | \/ |  | \      / |
                    |  \    /  |
                    |   \  /   |
                    |    \/    |

N=0  N=1    N=2     N=5
```

```
public void printPattern( )
{
   /* Answer: */
} // end of printPattern()
```

8. Given 2 USB storage devices, each of capacity m, and 6 computer files of sizes s1, s2, s3, s4, s5, s6, where s1 + s2 + s3 + s4 + s5 + s6 < 2m.

Write a method `public Boolean canPack(int[] s, int m)` to determine whether the 6 files can be stored into the 2 devices, where each file must be stored completely in one of the 2 devices. `canPack` accepts parameters `m` and an array s of integers for s1, s2, …, s6, and returns true if the files can be stored, and returns false otherwise. Also write any method that `canPack` calls. (20%)

Partial List of Java Operators in Decreasing Precedence	Associativity
`()` `++ (postfix)` `-- (postfix)`	left-to-right
`+ (unary)` `- (unary)` `++ (prefix)` `-- (prefix)` `!`	right-to-left
`*` `/` `%`	left-to-right
`+ (addition)` `- (subtraction)`	left-to-right
`<` `<=` `>` `>=`	left-to-right
`==` `!=`	left-to-right
`&&`	left-to-right
`||`	left-to-right
`?:`	right-to-left
`=` `+=` `-=` `*=` `/=` `etc.`	right-to-left
`, (comma operator)`	left-to-right

Part of the Unicode Table/ ASCII Table

0 NUL	1 SOH	2 STX	3 ETX	4 EOT	5 ENQ	6 ACK	7 BEL
8 BS	9 HT	10 NL	11 VT	12 NP	13 CR	14 SO	15 SI
16 DLE	17 DC1	18 DC2	19 DC3	20 DC4	21 NAK	22 SYN	23 ETB
24 CAN	25 EM	26 SUB	27 ESC	28 FS	29 GS	30 RS	31 US
32 SP	33 !	34 "	35 #	36 $	37 %	38 &	39 '
40 (	41)	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [	92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 {	124 |	125 }	126 ~	127 DEL

<END OF PAPER>