COE 212 – Engineering Programming

Welcome to Exam II
Monday May 13, 2013

Instructors: Dr. Randa Zakhour
Dr. Maurice Khabbaz
Dr. George Sakr
Dr. Wissam F. Fawaz

Name: ___Solution Key__________

Student ID: ________________

Instructions:

1. This exam is **Closed Book**. Please do not forget to write your name and ID on the first page.
2. You have exactly **140 minutes** to complete the **seven** required problems.
3. Read each problem carefully. If something appears ambiguous, please write your assumptions.
4. Do not get bogged-down on any one problem, you will have to work fast to complete this exam.
5. Put your answers in the space provided only. No other spaces will be graded or even looked at.

Good Luck!!

Problem 1: Multiple choice questions (**15 minutes)** [10 points]

For each question, choose the **single** correct answer.

1) Consider the program segment given below.
```
int i=0;
for(i=2; i < 5; i++)
     System.out.println(i);
```
   How many times does the body of the "for" loop get executed?
   a. 5
   b. 4
   c. 3
   d. None of the above
2) Which of the following is not a Java keyword?
   a. `break`
   b. `next`
   c. `while`
   d. None of the above
3) Which of the following statements about the conditional operator (`?:`) is false?
   a. It is a ternary operator, meaning that it takes three operands
   b. The first operand is a boolean expression
   c. The second operand is the resulting value if the condition evaluates to true
   d. The third operand is the resulting value if the condition evaluates to true
4) Which of the following is a proper way to call the method `readData` four times?
   a.
```
int i=0;
while(i<=4) {
     readData();
     i=i+1;
}
```
   b.
```
int i=0;
while(i<4) {
     readData();
}
```
   c.
```
int i = 9;
while(i!=4) {
     readData();
     i--;
}
```
   d. None of the above
5) Which of the following for loop headers results in equivalent numbers of iterations?
   A. `for(int q=1; q<=100; q++)`
   B. `for(int q=100; q >=0; q--)`
   C. `for(int q=99; q > 0; q -= 9)`
   D. `for(int q=990; q > 0; q -= 90)`
   a. A and B
   b. C and D
   c. A and B have equivalent iterations and C and D have equivalent iterations
   d. None of the loops have equivalent iterations
6) Which of the following code choices results in having a value for total that is equivalent to the one computed by the code segment given below?

```
int total = 0;
for(int i=0; i<=20; i+=2)
total += i;
```

a.
```
int total=0;
for(int i=2; i<=20; i++) {
      total += i;
      i++;
}
```
b.
```
int i=2, total = 0;
while(i<=20) {
      if(i%2 != 0) {
            i++;
            continue;
      }
      total+=i;
      i++;
}
```
c. All of the above
d. None of the above

7) For the code segment below:
```
switch(q) {
case 3:
     System.out.println("banana");
     break;
case 4:
     System.out.println("pear");
case 5:
     System.out.println("grapes");
default:
     System.out.println("kiwi");
}
```
Which of the following values for `q` will result in `kiwi` being included in the output?
a. Any value different from 3, 4, and 5
b. Anything greater than or equal to 4
c. All of the above
d. None of the above

8) Which of the following is **false** about `Math.PI`?
a. It is a static method defined in the `Math` class
b. It is a static data member of the `Math` class
c. It is a double constant value
d. None of the above is false

9) Which of the following is **false** about static variables?
a. Only one copy of a static variable is created
b. A static variable cannot be referenced by a static method
c. A static variable can be accessed through the name of the class
d. None of the above is false

10) Which of the following is not a `Scanner` method?
a. `hasNext()`
b. `nextLine()`
c. `nextToken()`
d. None of the above

<u>Problem 2:</u> True or false questions (**15 minutes**) [12 points]

1. The output of the program below is only: `Exiting`
```
boolean notDone = false;
int numberOfPieces = 4;
if(notDone)
if(numberOfPieces > 4) { System.out.println(“You won”); }
else { System.out.println(“Keep playing”); }
System.out.println(“Exiting”);
```
Answer: **True** False

2. The following code prints the words: `second Done`
```
char c = ‘b’;
switch(c) {
case ‘a’: System.out.print(“third”);
case ‘b’: System.out.print(“second”);
case ‘c’: System.out.print(“first”);
}
System.out.println(“ Done”);
```
Answer: True **False**

3. Consider the code fragment given in the box below, where `Piece`, `Board`, and `PlayGame` are all Java classes. Assume this code fragment is syntactically correct. The code shown in the box below would give the same result as the following statement:
`theColor=theBoard.getPieceAtIndex(5).getCircle().getColor();`
```
Board theBoard = new theBoard();
int x = 5;
Piece aPiece = theBoard.getPieceAtIndex(x);
Circle aCircle = aPiece.getCircle();
theColor = aCircle.getColor();
```
Answer: **True** False

4. The following code in Java stores in variable `sum` the sum of the positive odd integers less than 10.
```
int sum = 0;
int x;
for(x=-1; x<=10; x+=2)
      if(x>0) sum += x;
```
Answer: **True** False

5. The following is a valid declaration in Java: `Boolean If = true;`
Answer: **True** False

6. The contents of two strings can be compared using either (==) or `equals` method.
Answer: True **False**

7. Any code that can be written with multiple `if-else` statements can also be written with a `switch` statement.

Answer: True **False**

8. The output of the program segment given below is: `true`

```
String last = “first”;
if(last.equals(“first”))
      System.out.print(“last == first”);
```

Answer: True **False**

9. Consider the following sequence of `if` statements:

```
if(a < b)
  if(c < d)
      if(e < f)
            answer = e;
```

The series of `if` statements above can be rewritten as follows:

```
 if((a < b) || (c < d) || (e < f))
      answer = e;
```

Answer: True **False**

10. The output of the segment of code shown below is: `sum: 0`

```
int sum=-1;
for(sum=0; sum > 0; sum++)
      sum--;
 System.out.println(“sum: ” + sum);
```

Answer: **True** False

11. The following can be used to check to see if the value of variable x is between 10 (inclusive) and 20 (inclusive): `if(!(x<10 || x>20))`

Answer: **True** False

12. The following code prints the words: `Grade is: B`

```
int x = 87;
char result = ‘A’;
System.out.print(“Grade is: ”);
if(x > 90)
      result = ‘A’;
      if(x > 80)
            result = ‘B’;
            if(x > 70)
                  result = ‘C’;
                  if(x > 60)
                        result = ‘D’;
System.out.print(result);
```

Answer: True **False**

Problem 3: Code analysis (**20 minutes**) [12 points]

1) Assume that an instance of the class called `Compare` given below is created as follows: `Compare compareInstance = new Compare(4);`
What output is produced by this constructor call?

```
public class Compare {
      private int x, y;
      public Compare(int x) {
            this.x = x; y = 2;
            first();
      } // end constructor Compare
      public void first() {
            int n = 2;
            if(second(n, x) && second(y, n))
                  System.out.print(x + “,” + y);
            System.out.print(“Done”);
      } // end method first()
      public void second(int a, int b) {
            x = b/a;
            y = y-x;
            if(x < 3)
                  return true;
            else
                  return false;
      } // end method second()
} // end class Compare
```

a. `2,0 Done`
b. `0,2 Done`
c. `Done`
d. None of the above

2) Again consider the code for the `Compare` class given above. What output is produced if the following statement is executed:
`Compare compareInstance = new Compare(8);`

a. `2,0 Done`
b. `0,2 Done`
c. `Done`
d. None of the above

3) Consider the methods given below. What would be the output if it were called using the statement: `method1(35, 10);`?

a. 0
b. 10
c. 70
d. none of the above

```
public void method1(int x, int y){
      int temp = method2(x,y);
      System.out.print(x*y/temp);
}
public int method2(int x, int y) {
      while (x!= y)
            if (x > y)
                  x -= y;
            else
                  y -= x;
      return x;
}
```

<u>Problem 4:</u> Evaluating java expressions (**20 minutes)** [12 points]
For each of the following code fragments, what is the value of x after the statements are executed?

(1)
```
int x=0;
for(int i=0; i<3; i++)
   for( int j=i; j<3; j++)
      x = x + j;
```
Answer: **x = 8**

(2)
```
String x = “”;
boolean notDone = true;
if (notDone != false)
     x.concat(“Not”);
     x += “done”;
if(notDone == true)
     x += “End”;
```
Answer: **x = “NotdoneEnd”**

(3)
```
String x = “”;
int y = 1;
for(int i=0; i<4; i=y-i) {
     x += i;
     y += i;
}
```
Answer: **x = “01123”**

(4)
```
String coolName = "regit *nnhoj", x = "";
int i = 1;
while(i <= coolName.length()) {
if (coolName.charAt(i-1) == ' ') x += 'y';
else
     x += coolName.charAt(coolName.length() - i);
i++; }
```
Answer: **x = johnny tiger**

(5)
```
String s = new String("this is it");
int x = 0, y = 0;
do {
     if (s.charAt(y) == 'i')
          x++;
     y += 2;
} while (y < s.length());
```
Answer: **x = 2**

(6)
```
int x=3, i=1;
do { x %= i; i++;} while(i < 5);
x -= i;
```
Answer: **x = -5**

<u>Problem 5:</u> Debugging Problem (**10 minutes)** [6 points]
Consider the following program, which is supposed to print out an approximation of π calculated according to the following equation, where N is a non-negative integer that is entered by the user.

$$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots + (-1)^N \frac{1}{2N+1}\right)$$

```
1     import java.util.Scanner;
2     public class PiApproximationBuggy {
3          public static void main(String[] args){
4               Scanner scan = new Scanner(System.in);
5               int N = scan.nextInt();
6               int i = 0;
7               float pi = 0.0;
8               while (i <= N){
9                    if (i % 2 == 0)
10                        pi += 1/(2*i+1);
11                   else
12                        pi -= 1/(2*i+1);
13              i++;}
14              pi *= 4;
15              System.out.println("PI = " + pi);
16    }}
```

This program has three bugs.

A. Which bug prevents the program from *compiling* successfully? Identify the line number where the bug appears and give a correct version of this line of code.

Line number _**7**____

Correct version: __**`float pi = 0.0f;`**____________________

B. Identify the line numbers where the two *logical* errors appear and give a correct version of each line of code.

Line number _**10**____

Correct version:

__**`pi += 1.0/(2*i+1);`**______________________________________

Line number ___**12**__

Correct version: __**`pi -= 1.0/(2*i+1);`**__________________________

<u>Problem 6:</u> Method definition (**20 minutes)** [8 points]

You are given below the headers of 3 methods called **`extractInt`**, **`isPrime`**, **`reverseInt`** and **`containsEmirp`**. Your job is to complete the definition of each one of these methods as per the provided guidelines.

1. **`extractInt()`** is a method that accepts a `String` parameter, extracts all the digits contained inside this `String` and combines them together in the order of their occurrence resulting in a new integer. For example, if the entered `String` is "The 1st 4 prime numbers are 2, 3, 5 and 7." then the method would return the `int` 142357.

```
public int extractInt(String s){
     String result = “”; char d;
     String digits = “0123456789”;
     for (int i = 0; i < s.length(); i++){
          d = s.charAt(i);
          if (digits.indexOf(d) != -1)
               result += d;
     }
     return Integer.parseInt(result);
}
```

2. **`isPrime()`** is a method that accepts an `int` parameter and returns true if the number is prime, i.e. only divisible by itself and by 1.

```
public boolean isPrime(int n) {
    //check if n is a multiple of 2
          if (n % 2 == 0) return false;
    //if not, then just check the odds
          for(int i = 3; i*i <= n; i+=2) {
                if(n % i == 0)
                     return false;
          }
    return true;
}
```

3. **`reverseInt()`** is a method that takes as a parameter an `int` and returns a reversed version of that `int`.

   ```
   public int reverseInt(int original){
        int reversed = 0;
        while (original > 0)
             reversed = reversed * 10 + original % 10;
             original /= 10;
        return reversed;
   }
   ```

4. **`containsEmirp()`** is a method that accepts a `String` parameter, extracts an `int` from it using the **`extractInt`**() method defined earlier and returns true if both the extracted `int` and its reversal (i.e. the number written backwards obtained through the use of **`reverseInt()`**) are prime; you should also use the **`isPrime`**() method defined earlier. For example, if the `String` is "He1ll7o9", then the method will return `true` since the extracted `int` 179 and its reversal 971 are both prime numbers.

   ```
   public boolean containsEmirp(String s) {
        int original = extractInt(s);
        int reversed = reverseInt(original);
        return (isPrime(original) && isPrime(reversed));
   }
   ```

Problem 7: Coding (**40 minutes)** [40 points]

1. Write a program named `CountChars` that reads a String from the user and prints the number of vowels (the letters a, i, e, o and u, regardless of capitalization), the number of consonants (all other alphabetic characters) and the number of all non-alphabetic characters (digits, spaces, punctuation, etc.) found in the string.

Sample output

```
Enter a sentence: This is the exam for COE212
Vowel count:  8
Consonant Count: 11
All other characters Count: 8
```

```
import java.util.Scanner;

public class CountChars {
     public static void main (String[] args) {
          Scanner scan = new Scanner(System.in);
          String vowels = “aeiouAEIOU”;
          String consonants = “bcdfghjklmnopqrstvwxyz” +
                              “BCDFGHJKLMNOPQRSTVWXYZ”;
          int countVows = 0, countCons = 0, countOther = 0;
          char c;
          System.out.println(“Enter a string: “);
          String s = scan.nextLine();
          for (int j = 0; j < s.length(); j++){
               c = s.charAt(i);
               if (vowels.indexOf(c) > 0)
                    countVows++;
               else
                    if (consonants.indexOf(c) > 0)
                         countCons++;
                    else
                         countOther++;
          }
          System.out.println(“There are ” + countVows +
                          “ vowels, ” + countCons +
                          “ consonants and ” + countOther +
                          “ other characters.”);
     }
}
```

2. Write a program called `MilesToKM` that reads from the user a lower and an upper bound on the number of miles and an integer step size and produces a conversion table to kilometers (note that 1 mile is 1.609 miles) for all values between the lower (inclusive) and upper bound (inclusive), taking steps equal to the step size.

Sample output

```
Enter minimum number of miles: 5
Enter maximum number of miles: 30
Enter step size: 10

Miles      Kilometers
5          8.045
15         24.135
25         40.225
```

```
import java.util.Scanner;

public class MilesToKM {
   public static void main (String[] args){
          Scanner scan = new Scanner(System.in);
          System.out.print(“Enter a lower bound: ”);
          int lowerBound = scan.nextInt();
          System.out.print(“Enter an upper bound: ”);
          int upperBound = scan.nextInt();
          System.out.print(“Enter a step size: ”);
          int stepSize = scan.nextInt();
          int kilometers, miles;
          System.out.println(“Miles \t Kilometers”);
          for (int i = lowerBound; i <= upperBound;
                                   i+= lowerBound)
          {
               miles = i;
               kilometers = miles * 1.609;
               System.out.println(miles + “ \t ” +
                                  Kilometers);
          }
     }
}
```

3. Write a program that prints the numbers from 1 to 100 (inclusive). But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

```
public class FizzBuzz {
     public static void main (String[] args) {
          for (int i = 1; i <= 100; i++) {
               if(i % 3 == 0 && i % 5 !=0)
                    System.out.println(“Fizz”);
               if(i % 3 != 0 && i % 5 == 0)
                    System.out.println(“Buzz”);
               if(i % 3 == 0 && i % 5 == 0)
                    System.out.println(“FizzBuzz”);
               if(i % 3 != 0 && i % 5 != 0)
                    System.out.println(i);

          }
     }
}
```

4. Write a program to read the contents of a text file called `“test.txt”`. The program has to write the following to a file called `“fileStats.txt”`:
    1. The total number of words in the file.
    2. The number of times the word “the” (ignoring capitalization) occurred in the file.

```
import java.io.*;
import java.util.*;

public class ContentStats {
     public static void main (String[] args)
                             throws IOException{
          File f = new File(“test.txt”);
          File g = new File(“fileStats.txt”);
          FileWriter fw = new FileWriter(g);
          BufferedWriter bw = new BufferedWriter(fw);
          PrintWriter pw = new PrintWriter(bw);
          Scanner fileScan = new Scanner(f);
          int countWords = 0, countThe = 0;
          while (fileScan.hasNext()){
               String s = fileScan.nextLine();
               Scanner stringScan = new Scanner(s);
               while(stringScan.hasNext()){
                    countWrods++;
                    String word = stringScan.next();
                    if (word.equalsIgnoreCase(“the”))
                         countThe++;
               }
          }
          pw.println(“The number of words is: ” +
                                             countWords);
          pw.println(“The frequency of occurrence of ” +
                     “the word \“the\” is:” + countThe);
          pw.close();
     }
}
```