# COE 212 – Engineering Programming

Welcome to Exam II
Tuesday December 16, 2014

Instructors: Dr. Bachir Habib
Dr. Georges Sakr
Dr. Joe Tekli
Dr. Wissam F. Fawaz

**Name: _____**

**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 **125 minutes** to complete the 5 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) What is the output by the following Java code segment?

```
int temp = 80;
if(90 < temp <= 200)
System.out.println(“This room is too hot”);
if(temp <= 90)
System.out.println(“This room is too cold”);
if(temp == 80)
System.out.println(“This room is just right!”);
```

   a. This room is too hot
      This room is too cold
   b. This room is too cold
      This room is just right!
   c. This room is too hot
      This room is just right!
   d. **None of the above**

2) Which of the following statements about the conditional operator (`?:`) is **false**?
   a. The conditional operator is a ternary operator, meaning that it takes three operands
   b. The first operand is a `boolean` expression
   c. **The second operand is the result value if the condition evaluates to false**
   d. The third operand is the result value if the condition evaluates to false

3) What is output by the following Java code segment?

```
int temp = 180;
while(temp != 80) {
      if(temp > 90) {
            System.out.print(“This room is too hot! ”);
            temp = temp – (temp>150 ? 100 : 20);
      } else {
            if(temp < 70) {
                  System.out.print(“This room is too cold! ”);
                  temp = temp + (temp < 50 ? 30 : 20);}}
}
if(temp == 80) { System.out.println(“This room is just right!”);}
```

   a. This room is too cold! This room is just right!
   b. **This room is too hot! This room is just right!**
   c. This room is just right!
   d. None of the above

4) Which of the following segments is a proper way to call the method `readData()` four times?

   a.
```
int k = 0;
while(k != 4) {
      readData();
      k = k + 1;}
```

   b.
```
int k = 0;
while(k <= 4) {
      readData();
      k = k + 1;}
```

   c.
```
int k = 0;
while(k < 4) {
      readData();}
```

   d. None of the above

5) Which of the following for-loop headers result(s) 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 D
d. None of the loops have equivalent iterations

6) For the code segment below:

```
switch(q) {
      case 3:
            System.out.println(“apple”);
            break;
      case 4:
            System.out.println(“orange”);
            break;
      case 5:
            System.out.println(“grapes”);
      default:
            System.out.print(“kiwi”);
}
```

Which of the following values for `q` will result in `kiwi` being included in the output?

a. Any integer different than 3
b. Any integer strictly less than 3 and strictly greater than 5
c. 5
d. **Both (b) and (c)**

7) Consider the code segment below:

```
if(gender == 1)
      ++seniorFemales;
if(age >= 65)
      ++seniorFemales;
```

This segment is equivalent to which of the following?

a.
```
if(gender == 1 || age >= 65)
++seniorFemales;
```
b.
```
if(gender == 1 & age >= 65)
++seniorFemales;
```
c.
```
if(gender == 1 AND age >= 65)
++seniorFemales;
```
d. **None of the above**

8) Which of the following is equivalent to: `if(!(grade == sentinelValue))`?

a. **`if(grade < sentinelValue || grade > sentinelValue)`**
b. `if(grade !== sentinelValue)`
c. `!if(grade == sentinelValue)`
d. None of the above

9) Which of the following can be used to exit the body of a loop immediately and resume execution at the statement immediately following the loop?

a. `continue`
b. `return`
c. `exit`
d. **None of the above.**

10) Which of the following is an interface?

a. `Iterator`
b. `Comparable`
c. **Both of the above.**
d. None of the above.

## **Problem 2:** True or false questions (**20 minutes)** [12 points]

1. Code 1 and Code 2 given below are equivalent.

| **Code 1:** | **Code 2:** |
|---|---|
| `int i=0;`<br>`while (i < 20) {`<br>`  i++;`<br>`  System.out.println(i);`<br>`}` | `for (int i=0; i<=20; i++){`<br>`  System.out.print(i);`<br>`  System.out.println();`<br>`}` |

Answer: True **False**

2. Code 3 and Code 4 given below are equivalent.

| **Code 3:** | **Code 4:** |
|---|---|
| `int x=2, y=20, counter=0;`<br>`for(int j=y%x; j<100; j+=(y/x){`<br>`  counter++;`<br>`}`<br>`System.out.println(counter);` | `int counter = 0;`<br>`for(int j=10; j > 0; j--){`<br>`  ++counter;`<br>`}`<br>`System.out.println(counter);` |

Answer: **True** False

3. The output of the following nested for loops will be as shown in the box given below.

```
for (int i = 0; i < 4; i++) {
   for(int j=0; j < i; j++)
         System.out.print(" ");
   for (int k = 0; k< (4-i); k++)
         System.out.print("*");
   System.out.println();}
```

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

Answer: **True** False

4. The following code in Java prints all the odd numbers from 1 to 30 (inclusive), each value on a different line.

```
int i=0;
do {
      i++;
      System.out.print(i);
      if(i%2==0)
      System.out.println();
      i++;
} while(i <= 30);
```

Answer: True **False**

5. If a class implements `Iterator`, then an object created from that class can invoke the `hasNext` and `nextLine` methods.

Answer: True **False**

6. It is possible to use a static instance variable to count the number of objects created from a given class.

Answer: **True** False

7. The following code prints: `undone`

```
char c='a';
switch(c) {
case 'a': System.out.print('u');
case 'b': System.out.print('n');
default: System.out.print('d');
}
System.out.println('one');
```

Answer: True **False**

8. The following Java code segment can be used to swap the values of variables `a` and `b`

```
a = a + b;
b = a – b;
a = a – b;
```

Answer: **True** False

9. Consider the following `if` statement:

```
if(! ((a < b) && (c < d)))
      answer = 2;
```

The `if` statement above can be rewritten as follows:

```
 if((a >= b) || (c >= d))
      answer = 2;
```

Answer: **True** False

10. The output of the segment of code shown below is: `sum: 12`

```
int sum=0;
while(true) {
      if(sum <= 12)
            sum++;
      else
            break;
}
System.out.println("sum: " + sum);
```

Answer: True **False**

11. The following can be used to print out the greatest common divisor of `int` variables `x` and `y`:

```
while(x != y) {
      if(x < y)
            x = x –y;
      else
            y = y – x;
}
System.out.print(x);
```

Answer: True **False**

12. The following code can be used to print the characters of `String` variable `str` in reverse order.

```
for(int i=str.length(); i >= 0; i--)
   System.out.print(str.charAt(i));
```

Answer: True **False**

## **Problem 3:** Code analysis (**15 minutes**) [12 points]

1) Consider the methods given below. For positive parameters, how would you best describe its return value?

```
public int method1(int x, int y){
      int answer = 0;
      for(int i = 0; i < x; i++) {
            for(int j=0; j < y; j++) {
                  answer++;
            }
      }
      return answer;
}
```

a. `x+y`
b. `x*x`
**c. `x*y`**
d. Compile-time error
e. None of the above

2) Consider the `method2` shown below. For positive parameters, how would you best describe its return value?

```
public int method2(int x, int y){
      int answer = 0;
      while(x + y < x * y) {
            answer = answer + x;
            y = y-1;
            x = x+1;
      }
      return answer;
}
```

a. `x+y`
b. `x*x`
c. `x*y`
d. `x`
**e. None of the above**

3) What would the output of the method call: `method3("rail");`

```
public String method3(String aWord){
      char c,d;
      String word1 = "";
      String word2 = "";
      for( int i=aWord.length()/2; i>0; i--) {
            c = aWord.charAt( i);
            d = aWord.charAt( aWord.length() - i);
            word1 = word1 + c;
            word2 = d + word2;
      }
      return word1 + word2;
}
```

a. `liar`
b. `ilail`
**c. `iali`**
d. Compile-time error
e. None of the above

## **Problem 4:** Evaluating Java Expressions (**15 minutes**) [16 points]

For each of the following code fragments, what is the value of **x** after the statements are executed?

(1)
```
int x=0, i=2;
do {
x -= ++i;
i*=2;
} while(i < 5);
x -= i;
```

Answer: x =-9

(2)
```
String answer = "Sub"; int x = 10;
switch(answer) {
case "Add":
     x += 2;
     System.out.println(x);
     break;
case "Sub":
     x -= 2;
case "Mul":
     x *= 2;
     break;
case "Div":
     x /= 2;
default:
     x+=2;
}
```

Answer: x =16

(3)
```
boolean x=false;
for(int i=-4; i<0; ++i)
if(i%2 == 0)
x = (((2-i) <= 0) && x)&& true;
```

Answer: x =false

(4)
```
int x=1, d;
for(d=x; d<10; d*=2)
x-=d;
d++;
```

Answer: x = -14

(5)
```
int n=1000; int x=n, i=0;
while(x>0) {
     x=x/5; i++;}
for(int j=0; j<i; j++)
     n= n%2;
x=n;
```

Answer: x =0

(6)
```
int y = 100;
boolean x =(y%2 == 0 && y / 10 != 0) || (y / 5 == 0);
```

Answer: x = true

(7)
```
int x = 0;
for(int i=0; i <= 4; i++)
for(int j=0; j <= i; j++)
x += j++;
```

Answer x = 10

(8)
```
String S = "Test 101", x = "";
for(int i=1; i < S.length(); i+=3)
x += S.charAt(i);
```

Answer x = “e 1”

## **Problem 5:** Coding (**60 minutes**) [50 points]

1. Write a Java program called `PrimeDigits` that reads a positive integer value `val` from the user and then prints out the number of prime digits contained in `val`. Note that a prime number is a number that is divisible only by itself and by one.

   **Sample output**

   **Enter a positive int: 356**

   **Number of prime digits: 2**

```
import java.util.Scanner;

class primeDigits{

public static void main (String [] args){

int digit, val, count=0;

Scanner scan = new Scanner(System.in);

do{
   System.out.println("Enter an int:");
   val = scan.nextInt();
while(val < 0);   // Performing user input validation

while (val > 0)   // Looping to identifying each digit in the number
      {
         digit = val % 10;  // Identifying the last digit

           // Testing if the digit is a prime
           boolean Prime = true;
           int i=2;

           while(i<digit && Prime == true)
           {
               if(digit%i == 0)   Prime = false;
               i++;
           }
             // End prime test

             if(Prime) count++;

         val = val / 10;
      }

   System.out.println("Number of prime digits:" + count);

  }
}
```

2. Write a Java program called `StringMutation` that reads a String `str1` from the user and then produces a new String `str2` that is identical to `str1` except that all the vowels are converted into uppercase and all the consonants are converted into lowercase. Note that the vowels in English are the following: *a*, *e*, *o*, *u*, and *i*.

   **Sample output**

   **Enter a string: Exam is fun**

   **Mutated string: ExAm Is fUn**

```
import java.util.Scanner;

class StringMutation{

public static void main (String [] args){


Scanner scan = new Scanner(System.in);

System.out.println("Enter a string:");
String S1 = scan.nextLine();

String vowels = new String("aoieu");
String S2 = "";

for(int i=0; i < S1.length(); i++) // Looping to process each character
{
      if (vowels.indexOf(S1.charAt(i)) != -1)
            S2 = S2 + S1.substring(i, i+1).toUpperCase();
      else
            S2 = S2 + S1.substring(i, i+1).toLowerCase();
}

        System.out.println("Mutated string:" + S2);

 }
}
```

3. We wish to package `n` eggs into `nb` boxes that can accommodate 12 eggs each. Write a Java program called `Packaging` that reads from the user the number of eggs `n,` and determines, **without using the division operator**, the number of boxes `nb` (completely full and ready to seal), as well as the number of remaining unpackaged eggs (i.e., the eggs that are left and not enough to fill a whole box).
   **Sample output**
   **Enter the number of eggs: 25**
   **Number of boxes: 2**
   **Remaining number of free eggs: 1**

```
import java.util.Scanner;

class Packaging{

public static void main (String [] args){

int n, nb=0, r=0;
Scanner scan = new Scanner(System.in);

do{
   System.out.println("Enter the number of eggs:");
   n = scan.nextInt();
} while (n <0);  // Performing user input validation


while(n > 12)    // Simulating division using a series of subtractions
{
      n -= 12;
      nb++;

}

System.out.println("Number of boxes:" + nb);
System.out.println("Remaining number of free eggs:" + n);

 }
}
```

4. Write a program called `Guessing` that implements a *guessing game* allowing the user (human player) to guess an integer number between 1 and 100, which is randomly generated by the program. The program answers by displaying `Too Big!` or `Too Small!` regarding each candidate number provided by the player until the player guesses the right number, at which the program displays `Bravo!` The player is allowed a maximum of 5 guesses, after which the program displays: `Looser! The right number was:` *number*

   **Sample output**

   **Enter guess n# 1: 25**
   **Too Small! Enter guess n#2: 70**
   **Too Small! Enter guess n# 3: 80**
   **Too Big! Enter guess n# 4: 75**
   **Too Small! Enter guess n# 5: 78**
   **Looser! The right number was: 77**

```
import java.util.Scanner;
import java.util.Random;

class Guessing{

public static void main (String [] args){

int n, N, nbGuesses=0;
boolean rightGuess = false;

Random rand = new Random();
N = rand.nextInt(100) + 1;

Scanner scan = new Scanner(System.in);

while (nbGuesses <5 && !rightGuess)
{
      nbGuesses++;

      System.out.print("Enter guess n#" + nbGuesses + ": ");
      n = scan.nextInt();

      if(n == N) rightGuess = true;
      else if(n < N) System.out.print("Too Small!");
           else System.out.print("Too Big!");
}

if(rightGuess) System.out.println("Bravo!");
else System.out.println("Looser! The right number was: " + N);


 }
}
```

5. Write a Java program called `ArmstrongNumber` that reads from the user a positive `int` value called `val`. The program should then print a message indicating whether or not the input value is an Armstrong number. Note that a value `val` made up of *n* digits is considered to be an Armstrong number if the sum of its individual digits raised to the $n^{th}$ power matches `val`. For example, 371 that is *n*=3 digits long is an Armstrong number since: $(3)^3 + (7)^3 + (1)^3 = 371$. Moreover, 1634 that is *n*=4 digits long is an Armstrong number since: $(1)^4 + (6)^4 + (3)^4 + (4)^4 = 1634$.

   **Sample output**

   **Enter an int: 372**

   **372 is not an Armstrong number**

```
import java.util.Scanner;

class ArmstrongNumber{

public static void main (String [] args){

Scanner scan = new Scanner(System.in);

int n;
do{
      System.out.print("Enter an int:");
      n = scan.nextInt();
} while(n < 0);    // Performing user input validation

String S = "" + n;   // equivalent to: S= Integer.toString(n)
                     // Processing the interger as a String
                     // makes it easier to process individual
                     // digits in this program

int sum = 0;
for(int i=0; i< S.length(); i++)
{
      sum += Math.pow(Integer.parseInt(S.charAt(i)+""), S.length());

}

if(sum == n) System.out.println(n +" is an Armstrong number");
else System.out.println(n +" is not an Armstrong number");

 }
}
```