Java
Practice Midterm

Name -
Period -

Circle your answers & fill out the Scantron form if provided. You may use the Java Quick Reference at http://apcentral.collegeboard.com/apc/public/repository/ap_comp_sci_a_quick_reference.pdf

Part I - Multiple Choice

1. Consider the following method:

```
public static double solve(int n)
{
   double answer = 1.0;
   for (int k = 1; k < n; k++)
      answer *= (double) k/(k + 1);
   return answer;
}
```

What value is returned by a call to method `solve`?

a. 0.0
b. the kth root of n
c. 1/n
d. 1/(n - 1)
e. $(1/2)^n$

2. Suppose the method `int sign(int x)` returns 1 if x is positive, -1 if x is negative, and 0 if x is 0. Given

```
int[] nums = {-2, -1, 0, 1, 2};
```

what are the values of the elements of `nums` after the following code is executed?

```
for (int k = 0; k < nums.length; k++)
{
   nums[k] -= sign(nums[k]);
   nums[k] += sign(nums[k]);
}
```

a. `-2, -1, 0, 1, 2`
b. `-1, 0, 0, 0, 1`
c. `0, 0, 0, 0, 0`
d. `-2, 0, 0, 0, 2`
e. `-2, 0, 0, 2, 3`

3. Consider the following code segment:

```
int x = 2;
int y = 4;
x += y;
y += x;
System.out.println("x = " + x + ", y = " + y);
```

What output is produced by executing this code?

a. `x = 2, y = 4`
b. `x = 6, y = 6`
c. `x = 6, y = 10`
d. `x = 6, y = 12`
e. `x = 24, y = 42`

4. Consider the following method:

```
public boolean evaluate(boolean p, boolean q)
{
   return ((p && q) || !(p || q));
}
```

What is returned by evaluate?

a. evaluate always returns `false`.
b. evaluate returns `true` when p and q are both true or both `false`.
c. evaluate returns `true` only when p and q are both `true`.
d. evaluate returns `true` only when p and q are both `false`.
e. evaluate returns `true` when either p or q is true and the other is `false`.

5. Consider the following method, which is intended to count the number of times a value is doubled before it is at least as large as a target value:

```
// precondition: d > 0
public void doubleUp(double d)
{
   static final double TARGET = 100.0;
   int count;
   while (d < TARGET)
   {
      count = 0;
      d *= 2;
      count++;
   }
   System.out.println(count);
}
```

Of the following, which best describes the error in `doubleUp`?

a. The `static` modifier cannot be used with a constant.
b. A variable is used without being initialized.
c. A `double` cannot be multiplied by an `int`.
d. The counter is incremented in the wrong place.
e. A variable is initialized in the wrong place.

6. A programmer has mistakenly placed a semicolon at the end of the while statement in the following code segment. What will be the result of the executing code?

```
while ( j < n );  // mistaken semicolon
{
   System.out.println("in loop");
   j++;
}
```

a. The code will behave in the same manner as if the semicolon were not included.
b. The machine will get caught in an infinite loop, appearing to do nothing.
c. The machine will get caught in an infinite loop, repeatedly printing the message, "in loop".
d. Depending upon the code that precedes this loop within the program, either an infinite loop will result or the "in loop" message will be printed a single time.
e. The code will not compile because of the presence of the semicolon.

7. Assuming that `b` is a `boolean` variable, what will be the result of executing the code?

```
b = (b != b);
```

a. Sets `b` to false.
b. Sets `b` to true.
c. Changes the value of `b` to its negation.
d. Leaves the value of `b` unchanged.
e. The code will not compile.

8. Consider the following method:

```
public static int mystery(int n)
{
   int a, b, c = 0;
   while (n > 0)
   {
      a = n / 10;
      b = n % 10;
      c *= 10;
      c += b;
      n = a;
   }
   return c;
}
```

Which statement best describes the value that is returned by `mystery`?

a. Ten to the n^{th} power is returned.
b. The number of digits in `n` is returned.
c. The number of times 10 is a factor of `n` is returned.
d. The number whose digits are reversed of `n` is returned.
e. The sum of the digits in `n` is returned.

9. Consider the following class declaration:

```
public class SimpleInt
{
   private int myInt;

   public void setVal(int v)
   {
      myInt = v;
   }

   public int getVal()
   {
      return myInt;
   }
}
```

Assuming that `x` is a properly constructed `SimpleInt`, which of the following statements will compile in a program that uses a `SimpleInt`?

I. `System.out.println(x.getVal());`
II. `System.out.println(x.myInt);`
III. `System.out.println(x.setVal(10));`

a. I only
b. II only
c. I and II only
c. I and II only
d. I and III only

10. Which of the following cannot always be checked by a compiler?

a. The version of a method being invoked.
b. The spelling of keywords.
c. Possible missing punctuation within code.
d. Return values matching method return types.
e. Invocation of a private method from outside the class.

11. Consider the following code segment.

```
public static int calculate(int x)
{
   x = x * x;
   x = x * x;
   return x;
}
```

What is the result of a call to compute?
a. Returns 2y
b. Returns x^4
c. Returns $2x^2$
d. Returns $4x^2$
f. Returns the original value of x, since the method is static

12. Which of the following is the negation of the expression `(a && !b)` ?
a. `!a || b`
b. `!(!a || b)`
c. `!a && !b`
d. `!(!a && b)`
e. `a && b || !b`

13. Consider the method `doubleUp` that is intended to return the number of times value must be doubled before it is greater than target. `doubleUp` does not work as intended.

```
public static int doubleUp(int value, double target)
{
   int value = 1;
   int count = 0;

   while (value < target)
   {
      count++;
      value *= 2;
   }
   return count;
}
```

Of the following, which best describes the error in `doubleUp`?
a. An extra local variable declaration statement is used.
b. An uninitialized variable is used.
c. An `int` cannot be compared to a `double`.
d. A variable is incremented in the wrong place.
e. The loop test (i.e. control expression) is incorrect.

14. Suppose that a `Widget` named `w` has been properly constructed in a client program. Which of the following expressions will compile and execute as intended without error? Assume that `myName` and `myCost` are instance fields.
a. `System.out.println(myName + " " + myCost);`
b. `System.out.println(w.myName + " "+ w.myCost);`
c. `System.out.println(w.myName + " " + (String) w.myCost);`
d. `System.out.println(w.getName() + " " + w.getCost());`
e. `System.out.println(w.getName() + " " + (String) w.getCost());`

The next two questions refer to the following information.
Consider the following code segment that is intended to output a message describing a student's academic status.

```
if (grade >= 90)
   System.out.print("Excellent");
if (grade >= 80)
   System.out.print("Good");
if (grade >= 70)
   System.out.print("Passing");
if (grade >= 60)
   System.out.print("Borderline");
if (grade < 60)
   System.out.print("Failing");
```

15. If the above code is executed with a grade of 80, what will be printed?

a. `ExcellentGood`
b. `Good`
c. `GoodPassingBorderline`
d. `GoodPassingBorderlineFailing`
e. `PassingBorderline`

16. The code segment is flawed. If the code is rewritten, which set of values below is best for testing its correctness?

a. {95, 85, 75, 65, 55}
b. {90, 89, 80, 79, 70, 69, 60, 59}
c. {100, 90, 80, 70, 60, 50, 0}
d. Ten values chosen randomly from 0 to 100 should be tested
e. {99, 89, 79, 69, 59, 49} plus several random values

17. What is always true when a `while` loop terminates (in the absence of a `break` statement)?

a. The loop control expression is `true`.
b. The loop control expression is `false`.
c. The variables used in the loop are undefined.
d. The variables used in the loop are redefined as 0 or `null`.
e. The body of the loop has been executed.

18. Consider the following code segment:

```
String[][] m = new String[6][3];

for (int k = 0; k < m.length; k++)
   m[k][m[0].length - 1] = "*";
```

Which of the following best describes the result when this code segment is executed?

a. All elements in the first row of `m` are set to `"*"`
b. All elements in the last row of `m` are set to `"*"`
c. All elements in the last column of `m` are set to `"*"`
d. The code has no effect.
e. `ArrayIndexOutOfBoundsException` error occurs

19. Consider the following code segment that is intended to toggle variable `x` between 1 and -1. Assume `x` initially contains either the value 1 or -1.

```
if (x == 1)
   x = -1;
else
   x = 1;
```

Which of the following statements performs the same logic as the code segment above?

a. `x += -x;`
b. `x -= -x;`
c. `x *= -x;`
d. `x /= -x;`
e. `x = -x;`

20. Which statement is legal?

a. `String s = "40";`
b. `Integer i = 40;`
c. `String s = new String("40");`
d. `Integer i = new Integer(40);`
e. All of the above are legal

Part II - Free Response FILL IN ALL BLANKS BELOW OR place your answers on lined paper

1. Write a method named `remove` as started below. The method finds the first occurrence of a specified `String` named `word` in a given `String` named `text` and returns `text` with `word` removed. If `word` is not found, the method returns `text` unchanged.

```
// precondition: text is a non-empty string that consists of words separated by
// single blank spaces
// postcondition: if word is found in text, its first occurrence is removed from text
// and text is then returned; otherwise text is returned unchanged
// Example: if text = "The cow jumped over the moon"
//          and word = "cow" then
//          "The  jumped over the moon" will be returned where there are 2 blank
//          spaces between the words "The" and "jumped"

public static String remove(String text, String word)
{

      if (text.indexOf(word) != -1)
      {
       String firstPart = text.substring(0, text.indexOf(word));

       String lastPart = text.substring(________1________ + ____2______.length());

       return firstPart + lastPart;
      }

      return text;
}
```

2. Assume that you are given the following `Sentence` class which has a class invariant that `myWords` will always contain a set of English words that are separated by single blank spaces. Each word in `myWords` only contains letters. That is, there are no punctuation symbols (such as periods, question marks, or even hyphens) or digits within `myWords`. There is no leading blank space in front of the first word in `myWords` and there is no trailing blank space behind the last word in `myWords`. You are also guaranteed as a class invariant that `myWords` will always contain at least 2 or more words.

```
public class Sentence
{
      private String myWords;             // instance field

      public Sentence()                   // default constructor
      {
            myWords = "I love Java";
      }

      public Sentence(String words)       // other constructor
      {
            ___________3___________  = ___________4____________ ;
      }

      public String getWords()            // accessor
      {
            return _____________5_____________ ;
      }
```

```
// a
// postcondition: the number of separate words in myWords is returned
public int numWords()
{
      int count = 0;

      // counting the total number of spaces and adding one
      for (int i = 0; i < __________6___________ ; i++)
      {
            if (myWords.substring(____7_____, ______8_______ ).equals(" "))
            {
                  ____________9_____________ ;
            }
      }

      return count + 1;
}

// b
// postcondition: the number of separate letters in myWords is returned
public int numLetters()
{
      // substracting the # of spaces from the total number of characters
      // HINT: use the method numWords that you implemented above

      return myWords.length() - (_____________10_____________ - 1);
}

// c
// precondition: key will not be a null string and it will be a word
//    that only contains letters
// postcondition: if the String key is found anywhere within myWords,
//    the boolean value true will be returned;
//    otherwise false will be returned.
public boolean find(String key)
{
      if (myWords.indexOf(key) >= ________11___________ )
      {
            return _________12__________ ;
      }

      return __________13___________ ;
}

// d
// precondition: num > 0 and num will be less than or equal to
//    the number of words in myWords
// postcondition: return the num^th word in myWords where the very
//    first word would be returned if num is 1
public String getWord(int num)
{
      if (num == 1)
      {
            return myWords.substring(0, myWords.indexOf(" "));
      }

      int start = myWords.indexOf(" ") + 1;
      int end = myWords.length();
      int count = 1;
```

```
      for (int i = myWords.indexOf(" ") + 1; i < myWords.length() - 1; i++)
      {
            if (myWords.substring(i, i + 1).equals(" "))
            {

                  count++;

                  if (count == num - 1)
                  {
                        start = i + 1;
                  }

                  if (count == num)
                  {
                        end = i;
                        break;
                  }
            }
      }

      return myWords.substring(__________14__________, _________15_________);
}

// e
// postcondition: return the boolean value true if any word
//    occurs twice or more in myWords
public boolean containsDouble()
{
      for (int i = 1; i <= numWords(); i++)
      {
            for (int j = _______16__________ ; j <= numWords(); j++)
            {
                  if (getWord(j).equals(getWord(i)))
                  {
                        return true;
                  }
            }
      }

      _______________17____________________;

}

// f
// postcondition: return the boolean value true if any word
//    ends with the letter lowercase 's'
public boolean containsWordThatEndsWithS()
{
      for (int i = 1; i <= ___________18___________ ; i++)
      {
            if (getWord(i).substring(getWord(i).length()-1).equals("s"))
            {
                  return true;
            }
      }
```

```
            return false;
      }

}



public class SentenceTest
{
      public static void main(String[] args)
      {
            Sentence test = new Sentence("I love Macintosh computers NOT");
            System.out.println("myWords: " + test.getWords());
            System.out.println("A. numWords: " + test.numWords());
            System.out.println("B. numLetters: " + test.numLetters());
            System.out.println("C. find(love): " + test.find("love"));
            System.out.println("D. getWord(3): " + test.getWord(3));
            System.out.println("E. containsDouble: " + test.containsDouble());
            System.out.println("F. containsWordThatEndsWithS: " +
                        test.containsWordThatEndsWithS());
      }
}
```