**First Name**: _____ **Last Name**: _____

**McGill ID**: _____ **Section**: _____

# Faculty of Science
# COMP-202A - Foundations of Computing (Fall 2017) - All Sections
# Midterm Examination

October 18th, 2017
18:00 - 21:00

Examiners: David Becerra Romero [Section 1 (MWF 8:35-9:25)]
Giulia Alberini [Section 2 (TR 11:35-12:55)]
Giulia Alberini [Section 3 (MWF 13:35-14:25)]

## Instructions:

- **DO NOT TURN THIS PAGE UNTIL INSTRUCTED**

- This is a **closed book** examination; only a legal-sized (8.5" by 14") **crib sheet** is permitted. This crib sheet can be single or double-sided; it can be handwritten or typed. Non-electronic translation dictionaries are permitted, but instructors and invigilators reserve the right to inspect them at any time during the examination.
- Besides the above, only writing implements (pens, pencils, erasers, pencil sharpeners, etc.) are allowed. The possession of any other tools or devices is prohibited.
- Answer **all** questions on the answer sheet.
- This examination has **11** pages including this cover page, and is printed on both sides of the paper. On page 10, you will find information about **useful classes and methods**. **You may detach the Appendix (page 10 onwards) from the examination if you wish.**
- **Please hand in ONLY your answer sheet**.

### Scoring

The exam will be scored as follows:

1. Questions 1 to 8 are worth 1 point each
2. Questions 9 to 24 are worth 2 points each
3. Question 25 is worth 40 points
4. Total: 80 points

Version-0

## True/False Section (1 point each)

1. `String` and `char` are two primitive data types.

2. What is the value of `b` when `x` is a variable of type `int` with value 2, and `a` is a variable of type `boolean` with value `false`.

```
boolean b = 4.0*(1/x) > 0 && ((!a && x<=5) || (!(3/x < 1)));
```

3. The following is a valid (the program would compile and run) way of printing out the characters of the `String s` one by one.

```
int i = s.length();
while(i>0) {
    System.out.println(s.charAt(i));
    i--;
}
System.out.println(s.charAt(i));
```

4. The following code creates a compile-time error

```
int[]x = new int[5];
for(int i=0;i<=x.length;i++){
   x[i] = i;
}
```

5. The decimal representation of the binary number 11011110 is 232

6. If you do not specify the return type for a method/function, java will by default return a String.

7. The following is a valid way of initializing the variable `number`.

```
public class Example {
   public static void main(String[] args) {
      int number = test();
   }
   public static int test(){
      return 3;
   }
}
```

8. The following code prints *3811*

```
System.out.println(3+8+"11");
```

## Short Answer & Multiple Choice (2 points each)

9. What prints when the following code executes?

```
int x = 3;
double y = (32%x)*(x/2);
System.out.println("The result is: " + y + y);
```

10. Which variables exist on the line marked ****HERE****?

```
public static void myMethod(String s) {
    int x = s.length();
    for(int i = 0; i<x; i++) {
        if(s.charAt(i) == 'z') {
            char c = s.charAt(i);
        } else {
            ****HERE****
            char d = (char)(s.charAt(i) + 1);
        }
    }
}
```

11. What prints when the following code executes?

```
public static void main(String[] args) {
   int i = 71;
   double d = 67.235;
   char c = 'A';
   c = (char) i;
   System.out.print(" c = " + c + ", ");
   i = 'A';
   System.out.print(" i = " + i + ", ");
   i = (int) d;
   System.out.print(" i2 = " + i + ", ");
   d = (char)i;
   System.out.print(" d = " + d);
}
```

12. What prints when the following code is run?

```
public class Test {
   public static void main(String[] args){
      int num = 10;
      increment(num);
      System.out.print(num);
   }
   public static int increment(int x) {
       x++;
       return ++x;
   }
}
```

13. Write a method header using the following description. Do not write the whole method!
The name of the method is `averageGreater`. The method loops through the input parameter `array`, which is an array of doubles. Then, it finds the average of the values in the array, and returns true if the average is greater than the integer input parameter `threshold`, false otherwise. The order of the inputs does not matter.

14. Consider the following method header:

```
public static int myMethod(double x, int y)
```

which of the following is a *valid* (the program would compile and run) way to use this method? Write the letters corresponding to the correct answer(s) on your answer sheet.

A
```
int i = 0;
double x = myMethod(i,i);
```

B
```
System.out.println(myMethod(2, 2.0/3));
```

C
```
int x = myMethod(myMethod(2.0,1/2),5);
```

D
```
double x = 2.0 * myMethod(Math.sqrt(16),'a');
```

E
```
String s = myMethod(3.5, 2);
```

15. What prints when the following code executes?

```
public class Example {
   public static void main(String[] args) {
      int counter = 1;
      method1(counter);
   }
   public static void method1(int counter){
      counter++;
      counter = method3(counter);
      method2(counter);
   }
   public static void method2(int counter){
      System.out.print(counter+" ");
   }
   public static int method3(int counter){
      counter++;
      method2(counter);
      counter+=2;
      return counter;
   }
}
```

16. What prints when the following method executes?

```
public static void newMethod() {
    String s = "cat";
    int x = 0;
    int y = 1;
    if(s.length()<4) {
        System.out.print(s.charAt(x));
        x++;
    }
    if(x==y) {
        System.out.print(s.charAt(x));
        s = s + " and ";
        x++;
    } else if(s.length()>4) {
        System.out.print(s.charAt(x));
    }
    System.out.println(x+y);
}
```

17. The following code reverses the order of elements inside the String array `a`. However, the range of the `for` loop is missing (see line 2). Please complete the code by replacing the strings "___" with the correct values. Make sure to write your answer on the answer sheet.

```
1   int n = a.length;
2   for (int i = ___; i < ___; i++) {
3      String temp = a[n-i-1];
4      a[n-i-1] = a[i];
5      a[i] = temp;
6   }
```

18. What will the following code print?

```
public static void main(String[] args) {
   int count = 0;
   for(int i=0; i<4; i++){
      count++;
      for(int j=0; j<i; j+=2){
         count++;
         for(int k=j; k<i; k++){
            count++;
         }
      }
   }
   System.out.println(count);
}
```

19. What prints when the following code executes?

```
for(int i = 2; i >=0; i--) {
    if(i==0) {
        for(int j =0; j <5; j++) {
            System.out.print('+');
        }
    } else if (i==1) {
        for(int j=0; j<5; j++) {
            if(j==0||j==4) {
                System.out.print('_');
            } else {
                System.out.print('+');
            }
        }
    } else {
        for(int j=0;j<5;j++) {
            if(j==2) {
                System.out.print('+');
            } else {
                System.out.print('_');
            }
        }
    }
    System.out.println();
}
```

20. The following code contains three errors (one compile time, one running time and one logical error). Please list the line of each error specifying its type.

```
1   public static void main(String[] args) {
2   double size = 5.7;
3   int counter = 0;
4   int[] arr = new int[(int)size];
5   for(int i=arr.length();i>0;i--)
6      arr[i-1] = (int)size/i-1;
7      counter++;
8   System.out.println("The number of iterations of for loop is:" + counter);
9   }
```

21. Which of the following code snippets will cause a compile-time error? Write the letters corresponding to the correct answer(s) on your answer sheet.

A
```
int[] numbers = new int[5];
for(int i=0; i<=numbers.length; i++) {
    numbers[i] = i;
}
```

B
```
char letter = 'a';
char nextLetter = letter + 1;
```

C
```
String s = "cats and dogs";
for(i=0, i<s.length(), i++) {
    System.out.print(s.charAt(i));
}
```

D
```
int x = 1/2;
double y = 10.0;
System.out.println(y/x);
```

E
```
String catNames = {"Kitty", "Tiger", "Ginger"};
```

22. What are the elements of `array1` when the following code is run?

```
public static void main(String[] args) {
   int []array1 = {1,2,3};
   int []array2 = {1,2,3};
   if(array1 == array2){
      array1[0] = array2[array2[1]];
   }else{
      array1[0] = array2[array1[0]];
   }
}
```

23. What are the elements of `x` after the following code executes?

```
public class Test {
    public static void main(String[] args) {
        int[] x = {2,4,6,8};
        newMethod(x);
    }

    public static int[] newMethod(int[] x) {
        int[] y = new int[x.length];
        for(int i = 0; i < x.length; i++) {
            y[i] = x[i]%2 + i;
        }
        return y;
    }
}
```

24. What prints when the following code is run?

```
public class Example {
   public static void main(String[] args) {
      int[] a = {1,2,3,4,5};
      swap(a[1],a[4]);
      System.out.println(a[0]+" "+a[1]+" "+a[2]+" "+a[3]+" "+a[4]);
   }
   public static void swap(int x, int y){
      int temp =x;
      x = y;
      y = temp;
   }
}
```

# Long Answer Question (40 points total)

Pig Latin is a language game in which regular English words are changed and obfuscated according to a number of simple rules, so that the result text becomes hard to understand. An example of a sentence in Pig Latin is

```
orfay isthay uestionqay, ouyay illway itewray omesay odecay atthay anslatestray
englishway intoway igpay atinlay.
```

which translates into

```
for this question, you will write some code that translates english into pig
latin.
```

Here are the rules to translate an English word into Pig Latin:

- Step 1:
  - If a word starts with one or more consonants, the consonant(s) are moved at the end of the word. So, for instance, "pig" becomes "igp", and "strong" becomes "ongstr".
  - If a word starts with a vowel, a 'w' is added at the end of the word. So, for instance, "english" becomes "englishw".
- Step 2: Add the suffix "ay" to the result of the first step. So, for instance, "igp" becomes "igpay", "ongstr" becomes "ongstray", and "englishw" becomes "englishway".

Thus, using the above rules, the English words "pig", "strong", and "english" translate into Pig Latin as "igpay", "ongstray", and "englishway" respectively.

**NOTE.** For the rest of the question assume the following:

- The Strings contain no other character beside lower case letters of the English alphabet and the space character.
- No String begins or end with a space character.
- No String contains two or more consecutive space characters.
- No String is empty or null.
- You are not allowed to use any methods from the String class other than `charAt()` and `length()`.
- Comments are not required.

Hint: *if you feel you don't know how to implement one of the methods, don't spend too much time concentrating only on that part. Continue answering the other parts of the question assuming the method has been implemented and works correctly. You can come back to it later.*

Inside a class named `PigLatin`, write the following methods:

## Part A: isAVowel Method

Write the `isAVowel` method. This method takes a character as input and returns `true` if the character is a vowel, `false` otherwise. You can assume that the vowels are 'a', 'e', 'i', 'o', and 'u', and all other characters are consonants, including 'y'.

## Part B: numberOfConstants Method

Write the `numberOfConstants` method. This method takes a `String` as input and returns the number of consecutive constants that are at the beginning of the `String`. To get full marks, this method should use the method `isAVowel()` defined in part A. For example, `numberOfConstants("elephant")` returns `0`, while `numberOfConstants("squirrel")` returns `2`.

## Part C: numberOfWords Method

Write the `numberOfWords` method. This method takes a `String` as input and returns an integer representing the number of words appearing in the input String. For example, `numberOfWords("elephants")` returns `1`, while `numberOfWords("elephants and squirrel")` returns `3`.

## Part D: translateOneWord Method

Write the `translateOneWord` method. This method takes a `String` as input. If the `String` contains more than one word, then the method should print a useful error message to the user and return the empty `String`. Otherwise, the method returns a `String` containing the translation of the input into Pig Latin. To get full marks, this method should use `numberOfConstants()` from Part B as well as `numberOfWords()` from Part C. For example,

- `translateOneWord("elephant")` returns `"elephantway"`,
- `translateOneWord("squirrel")` returns `"uirrelsqay"`, and
- `translateOneWord("pigs and dogs")` returns `""`.

## Part E: tokenize Method

Write the `tokenize` method. This method takes a single `String` as input, and returns a `String` array containing the words in the sentence, assuming that the words are separated by a single space character. Remember that you cannot use any methods from the `String` class beside `charAt()` and `length()`. To get full marks, this method should use `numberOfWords()` from Part C. As an example, calling `tokenize` on the String `"the quick brown fox jumped over the lazy dog"` results in a `String` array containing { `"the", "quick", "brown", "fox", "jumped", "over", "the", "lazy", "dog"` }. Calling `tokenize` with input `"squirrel"`, should simply return the String array with `"squirrel"` as its only element.

## Part F: displayTranslation Method

Write the `displayTranslation` method. This method takes a `String` as input and prints out the translation of the input sentence into Pig Latin. To get full marks, this method should use `translateOneWord()` from Part D, and `tokenize()` from Part E. As an example, `displayTranslation` on input `"the quick brown fox jumped over the lazy dog"` should display `ethay uickqay ownbray oxfay umpedjay overway ethay azylay ogday`.

SUMMARY OF JAVA STANDARD LIBRARY METHODS FOR SELECTED CLASSES

- `Arrays` (package `java.util.Arrays` Methods:
  - `public static boolean equals(int[] a, int[] b)`: Returns true if the two specified arrays are equal to one another.
  - `public static boolean deepEquals(int[][] a, int[][] b)`: Returns true if the two specified arrays are deeply equal to one another.
  - `public static String deepToString(Object[] a)`: Returns a string representation of the "deep contents" of the specified array.
  - `public static String toString(Object[] a)`: Returns a string representation of the contents the specified array.
- `String` (package `java.lang`) Methods:
  - `public boolean equals(Object anObject)`: Compares this `String` to `anObject`.
  - `public int length()`: Calculates the length of this `String`.
  - `public char charAt(int i)`: Gets the `char` at position `i` of the `String`. Note that counting starts from 0 so that to get the first character of the `String` you should input `i` equals 0.
  - `public boolean equalsIgnoreCase(String anotherString)`: Compares, ignoring case considerations, this `String` to `anotherString`.
  - `public int compareTo(String anotherString)`: Compares this `String` to `anotherString` lexicographically; returns a negative value if this `String` occurs before `anotherString`, a positive value if this `String` occurs after `anotherString`, and `0` if both `String`s are equal.
  - `public int compareToIgnoreCase(String anotherString)`: Compares, ignoring case considerations, this `String` to `anotherString` lexicographically; returns a negative value if this `String` occurs before `anotherString`, a positive value if this `String` occurs after `anotherString`, and `0` if both `String`s are equal.
  - `public String substring(int start, int finish)`: Returns a new String composed of the `this` String starting from index `start` and up to, but not including index of `finish`
  - `public String replace(char c, char d)` : Returns a new String with all occurrences of the character `c` in the `this` String replaced by the character `d`.
  - `public char[] toCharArray()`: Converts this `String` to a new character array.
- `PrintStream` (package `java.io`) Methods:
  - `public void print(boolean b)`: Prints `boolean` value `b`.
  - `public void print(double d)`: Prints `double` value `d`.
  - `public void print(int i)`: Prints `int` value `i`.
  - `public void print(Object o)`: Prints `Object` `o`.
  - `public void print(String s)`: Prints `String` `s`.
  - `public void println()`: Terminates the current line by writing the line separator string.
  - `public void println(boolean b)`: Prints `boolean` value `b` and then terminates the line.
  - `public void println(double d)`: Prints `double` value `d` and then terminates the line.
  - `public void println(int i)`: Prints `int` value `i` and then terminates the line.
  - `public void println(Object o)`: Prints `Object` `o` and then terminates the line.
  - `public void println(String s)`: Prints `String` `s` and then terminates the line.
- `Math` (package `java.lang`) Methods:
  - `public static double pow(double a, double b)`: Returns the value of `a` raised to the power of `b`.
  - `public static double sqrt(double a)`: Returns the correctly rounded positive square root of `double` value `a`.
  - `public static double random()`: Returns a `double` value with a positive sign, greater than or equal to `0.0` and less than `1.0`.
  - `public static double exp(double a)`: Returns Euler's number $e$ raised to the power of `double` value `a`.

**ASCII Table**

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