Western University
Department of Computer Science

CS1027b Foundations of Computer Science II
Final Exam
May 4, 2017

Last Name: ______________________________________

First Name: ______________________________________

Student Number: ________________________________

Section Number (1 - Hughes, 2 - Solis-Oba): __________

Instructions

• Fill in your name, student number, and section.
• The exam is 3 hours long and it has a total of 115 marks; 15 marks are bonus marks.
• The exam has 12 pages and 29 questions.
• The first part of the exam consist of multiple choice questions. For each question circle only one answer.
• For the second part of the exam, answer each question only in the space provided.
• When you are done, raise your hand and one of the TA’s will collect your exam.
Part I. Multiple Choice Questions

For each multiple choice question circle only one answer.

1. (1 marks) Consider the following three Java classes

```
public class A
public class B extends A
public class C extends A
```

Consider the following statements

```
A varA = new B(); // Line 1
C varC = new C();
varA = varC;      // Line 3
```

Which line(s) generate compilation error(s)? (Line numbers are indicated in the comments).

(A) Line 1 (B) Line 3 (C) Lines 1 and 3 (D) None

2. (1 mark) Consider the same three classes A, B, and C from the previous question and the following Java statement

```
C varC = (C) new B();
```

Which of the following statements is true?

(A) The above Java statement does not generate any errors. (B) The above Java statement generates a compilation error. (C) The above Java statement generates a runtime error.

3. (1 mark) Consider the following Java code

```
int[] a = {3, 1, 2};
int[] b;
b = a;
b[0] = 2; // Line 1
System.out.println(a[0]+","+b[0]);
```

What does the above code print?

(A) 2,2 (B) 3,2 (C) Nothing. There will be a runtime error when executing line 1, as b has not yet been allocated memory.

4. (2 marks) Consider the following Java code

```
public class A {
    private int i = 1;
    public A() {
        int j = 0;
        for (int i = 0; i < 3; ++i) j = j + i;
        j = j + i;
        System.out.println("i = "+i+", j = "+j);
    }
}
```

What is printed when the statement

```
A varA = new A();
```

is executed?

(A) i = 1, j = 4 (B) i = 2, j = 5 (C) i = 3, j = 6 (D) Nothing is printed as the above code throws an exception.
5. (1 mark) Consider the following Java code

```java
Integer a = new Integer (2);
Integer b = new Integer (2);
if (a == b) m1(); // Line 1
else m2();
```

Which of the following statements is true?
(A) When the above code is executed, method m1() will be invoked.
(B) When the above code is executed, method m2() will be invoked.
(C) When the above code is executed, a compilation error will be generated on Line 1 as objects must be compared with the equals method.

6. (1 mark) Consider the following Java classes

```java
public class A {
    public boolean m() {return true;}
}

class B extends A {
    public boolean m() {return false;}
}
```

What does the following Java code print?

```java
A varA = new B(); // Line 1
if (varA.m()) System.out.println("true");
else System.out.println("false");
```

(A) false   (B) true   (C) Nothing. The program has a compilation error in Line 1.

7. (1 mark) The following values are inserted, in the given order, into an initially empty binary search tree: 13, 2, 9, 5, 23, 11.

Which value is stored in a node in level 3 of the tree? (Recall that the root is in level 0, the children of the root are in level 1, and so on.)

(A) 2   (B) 9   (C) 5   (D) 23

8. (1 mark) Consider a binary search tree T in which every node stores an integer value. Which of the following statements is always true?

(A) A preorder traversal visits the nodes in increasing order of the values stored in them.
(B) In an inorder traversal of T a node u storing value v is visited only after all nodes storing values smaller than v have been visited.
(C) The last node visited by a postorder traversal is the node storing the largest value.

9. (2 marks) Consider a sorted list implemented using a circular array. Let first be the index of the first element in the list and last be the index of the last element in the list. Let the list store n integer values, where n is an odd number. Operation median() returns the value of the median or the \(\frac{n+1}{2}\)-th smallest element in the list. For example, the median of this sorted list: 1, 4, 5, 8, 9, 12, 15, is 8. What is the time complexity of the best possible implementation of operation median()?

(A) O(1)
(B) O(log n)
(C) O(n)

10. (2 marks) Consider the same previous question, but this time the sorted list is implemented using a doubly liked list. Reference variable first points to the first node in the list and last points to the last node in the list. What is the time complexity of the best possible implementation of operation median()?

(A) O(1)
(B) O(log n)
(C) O(n)
11. (3 marks) Consider the following algorithm.

```java
public void sort(int[] a, int n) {
    ArrayQueue<Integer> q = new ArrayQueue<Integer>();
    for (int i = 0; i < n; ++i) {
        int x = i;
        (**)
        q.enqueue(a[x]);
        a[x] = -1;
    }
    for (int i = 0; i < n; ++i) a[i] = q.dequeue();
}
```

Which code must be inserted at the point marked (**) to sort array a in decreasing order? Array a stores n positive integer values.

(A) for (int j = i+1; j < n; ++j) if (a[j] < a[x]) x = j;
(B) for (int j = i+1; j < n; ++j) if (a[j] > a[x]) x = j;
(C) for (int j = 0; j < n; ++j) if (a[j] > a[x]) x = j;
(D) for (int j = 0; j < i-1; ++j) if (a[j] > a[x]) x = a[j];

12. (2 mark) Consider the following Java code.

```java
private static void m(int[] a) throws ArrayIndexOutOfBoundsException {
    try {
        for (int i = 0; i <= 3; ++i) a[i] = 0;
    }
    catch (NullPointerException e) {
        System.out.print("Error 1. ");
    }
}
```

public static void main(String[] args) {
    int[] b = {1, 2};
    try {
        m(b);
    }
    catch (Exception e) {
        System.out.println("Error 2. ");
    }
}

When the main method is executed what message(s) will be printed?

(A) Error 2.
(B) Error 1.
(C) Error 1. Error 2.
(D) Nothing will be printed.

13. (1 mark) Consider the following algorithm.

```java
private void m(int size) {
    if (size == 1) return;
    else m(size - 1);
}
```

How many activation records (or call frames) are created in the execution stack (or runtime stack, or call stack) for method m when the statement `m(3);` is executed?

(A) 0  (B) 2  (C) 3  (D) 4
14. (1 mark) Consider the following Java code.

```java
public class A {
    private int x;
    public String s = "";
    public A (int y) {
        x = y;
    }
}
```

Which of the following statements is correct?
(A) The empty string referenced by \( s \) is allocated memory in the heap, while \( x \), \( s \), and \( y \) are allocated memory in the execution stack (or runtime stack or call stack).
(B) The empty string referenced by \( s \) and \( s \) are allocated memory in the heap, while \( x \) and \( y \) are allocated memory in the execution stack.
(C) The empty string referenced by \( s \), \( s \), and \( x \) are allocated memory in the heap, while \( y \) is allocated memory in the execution stack.
(D) \( x \), and \( s \) are allocated memory in the heap; \( y \) is allocated memory in the execution stack. No memory needs to be allocated for the empty string.

15. (1 mark) Consider the following recursive definition:

\[
    f(n) = f(n-1) - f(n-2)
\]

\[
    f(1) = 2
\]

\[
    f(0) = 0
\]

What is the value of \( f(4) \)?
(A) -2  (B) 0  (C) 2  (D) 8

16. (2 marks) Consider the following Java code.

```java
public class classA<T> {
    public T smaller (T elem1, T elem2) {
        if (elem1.compareTo(elem2) < 0) return elem1; // Line 3
        else return elem2;
    }
}
```

Which of the following statements is correct?
(A) Line 1 has a compilation error.
(B) Line 3 has a compilation error.
(C) There are no compilation errors, but the above code will produce a runtime error when executed.
(D) There are no errors in this code.
Part II. Written Answers
Write your answers only in the space provided.

17. (3 marks) Consider the following algorithm.

```java
public int traverse(BinaryTreeNode r) {
    if (r == null) return 0;
    else {
        int v = traverse(r.getLeftChild()) - traverse(r.getRightChild());
        if (v > r.getElement()) return v;
        else return r.getElement();
    }
}
```

If the following statement is executed

```java
int result = traverse(r);
```

where \( r \) is the root of the tree on the right.

What value will \( \text{result} \) have? 

18. (5 marks) Consider the following Java code.

```java
public static void main (String[] args) {
    int[] a = new int[3];
    int size = 0;
    init(a, size);
    for (int i = 1; i < size; ++i) System.out.println(a[i]);
    System.out.println(a[0]);
}
```  

```java
private static void init(int[] a, int size) {
    for (int i = 0; i < 3; ++i) {
        ++size;
        a[i] = (1-i) * size;
    }
}
```

Show the output produced by this algorithm. Values must be printed in the same order as the algorithm prints them.

19. Consider the following Java code.

```java
public static void main(String[] args) {
    int[] a = {3, 5, 2};
    int result = m(a,0,2);
}
```  

```java
public static int m(int[] a, int first, int last) {
    if (first > last) return 1;
    else {
        int mid = (int)Math.floor((first+last)/2);
        return m(a,first,mid-1)+m(a,mid+1,last);// Line 1
    }
}
```

where method \( \text{Math.floor}(x) \) rounds the value of the parameter \( x \) down to the nearest integer. For example, \( \text{Math.floor}(3.2) = 3, \text{Math.floor}(3) = 3 \).

(3 marks) What value does \( \text{result} \) have at the end? 

(2 marks) How many recursive calls does the algorithm make (number of calls made in Line 1)? ___
Consider the implementation of a stack using a circular linked list. A circular linked list is a singly linked list where the last node of the list points to the first node of the list instead of to null. Each node of the list stores one element of the stack. There is a reference variable called `top` pointing to the node storing the value at the top of the stack and a reference variable called `bottom` pointing to the node storing the value at the bottom of the stack. For example, a stack storing values 9, 5, and 7, with 7 at the top and 9 at the bottom of the stack is represented with the circular linked list below.

Consider the following implementation of the `pop()` operation. Assume that elements stored in the stack are of type `T`; `top` and `bottom` are instance variables. Methods `getElement`, `getNext`, and `setNext` get the data stored in a node, get a reference to the next node in the list, and change the reference to the next node in the list, respectively.

```java
private T pop() {
    T element = top.getElement();
    top = top.getNext();
    bottom.setNext(top);
    return element;
}
```

Indicate whether this is a correct implementation of the `pop()` operation. If the implementation is incorrect, explain why it is incorrect by giving an example showing that the code will not perform the pop operation correctly. If the implementation is correct, explain why it always correctly removes and returns the element at the top of the stack.

21. (3 marks) Draw one binary tree in which every node stores one of the letters A, B, C, D, E and such that

- a preorder traversal visits the nodes in this order: E A D B C, and
- an inorder traversal visits the nodes in this order: A E B D C.
For each one of the following 4 questions, compute the time complexity of the given code. You must explain how you computed the time complexity and you must give the order (big-Oh) of the time complexity. (Hint. Your answer might be like this: “Number of operations performed outside the loops: x; number of operations performed in one iteration of the inner loop: y; number of iterations of the inner loop when i = 0 is $z_1$, when i = 1 is $z_2$, . . .; total number of operations performed by the loops: w. Total number of operations performed by the algorithm: x + w. The order of the time complexity is $O(v)$.”) The following fact might be useful to you: $m \sum_{k=1}^{m} = \frac{m(m+1)}{2}$.

22. (7 marks)

```c
int x = 0;
  for (int i = 0; i < n; i = i+1)
    for (int j = 0; j < n; j = j+1) {
      if (j < i) j = j + n;
      else x = x+1;
    }
```

23. (7 marks)

```c
int x = 1;
  for (int i = 0; i <= n * n; i = i + n)
    for (int j = 0; j < i; j = j + n)
      x = x + 1;
```
24. (7 marks)
    
    ```
    int j = 1;
    int i = 1;
    while (i <= n) {
        if (i == n)
            if (j < n) {
                i = 1;
                j = j+1;
            }
        else i = n+1;
        else i = i+1;
    }
    ```

25. (7 marks) In the following algorithm node r is the root of a binary tree with \( n > 0 \) nodes.
    
    ```
    public int traverse(BinaryTreeNode r) {
        int res = 2;
        if (r.getLeftChild() != null) res = res + traverse(r.getLeftChild());
        if (r.getRightChild() != null) res = res + traverse(r.getRightChild());
        return res;
    }
    ```
For the following 4 questions write algorithms in Java or in detailed Java-like pseudocode like the ones used in the lecture notes.

26. (12 marks) A binary tree is symmetric if for every internal node the number of nodes in its left subtree is the same as the number of nodes in its right subtree. Given a node p, let p.size() return the number of nodes in the subtree with root p, and p.getLeftChild() and p.getRightChild() return the left and right children of p, respectively. Write a recursive algorithm isSymmetric(r) that receives as parameter the root r of a binary tree and it returns true if the tree is symmetric and it returns false otherwise. For example for the tree below with root r the algorithm must return true, but for the tree with root s it must return false.

![Tree Diagram](image)

27. (8 marks) Let q be a queue storing n data items. Write an algorithm reverse (q) to reverse the queue so that the first element in q (the element at the front of the queue) becomes last (the element at the rear of the queue), the second element becomes the second last and so on. The only methods that you can use to manipulate the queue are q.enqueue(element) that adds an element to the rear of the queue, q.dequeue() that removes the element at the front of the queue, and q.isEmpty() that returns true if the queue is empty and it returns false if the queue is not empty. You cannot use any auxiliary data structures. (Hint. Design a recursive algorithm similar to the one in one of the questions of the midterm.)
28. (11 marks) Let \( q \) be a queue storing \( n \) integer values. Write an algorithm \( \text{partition}(q, \text{target}) \) that receives as parameter \( q \) and an integer value \( \text{target} \) and it re-arranges the values in the queue so that all the values smaller than \( \text{target} \) appear closer to the front of the queue than any values larger than or equal to \( \text{target} \). For this algorithm you can use one auxiliary stack \( s \). You cannot use any other auxiliary data structures. You can only use the following queue methods: \( q.\text{dequeue}() \), \( q.\text{enqueue}() \), \( q.\text{isEmpty} \) and \( q.\text{size}() \) (that returns the number of elements in the queue), and only the following stack methods: \( s.\text{push}() \), \( s.\text{pop}() \), and \( s.\text{isEmpty}() \).

For example, for the following queue on the left side and \( \text{target} = 6 \), your algorithm should produce a queue like the one on the right.

```
front  | 7  | 5  | 6  | 8  | 4  |
        |    |    |    |    |    |
rear   |    |    |    |    |    |
```

```
front  | 5  | 4  | 8  | 6  | 7  |
        |    |    |    |    |    |
rear   |    |    |    |    |    |
```
29. (11 marks) Write an algorithm `remove(target)` to remove the node storing a given value `target` from a sorted singly linked list. If no node in the list stores value `target` then the algorithm must return `null` otherwise it must return `target`. Instance variable `first` is a reference variable pointing to the first node in the list. If `current` is a reference to a node, `current.next()` returns a reference to the next node in the list (or `null` if `current` points to the last node in the list), `current.setNext(succ)` makes the node referenced by `current` point to node `succ`, and `current.getElement()` returns the data item stored in the node pointed by `current`. 