Part I. Multiple Choice Questions

For each multiple choice question circle **only one** answer. **Note.** In all questions that refer to singly linked lists or doubly linked lists, methods `getNext`, `getPrevious`, `setNext` and `setPrevious` are getter and setter methods to obtain and to set the next or previous node in the list.

1. (1 mark) Consider this class definition: `public class A<T> implements B<T>`. Class `A` must provide code to implement **every** method in the interface `B`.

   (A) True (B) False

2. (1 mark) An abstract data type (ADT) specifies a set of operations and **one** specific way to implement them

   (A) True (B) False

3. (1 mark) The following statement: `String[] s;` creates an array of `String` objects all of whose entries are null.

   (A) True (B) False

4. (1 mark) Consider the following Java code

```
private int m(int x, int y) {
   if (x != 0) return x;
   else return y/x;
}
```

   Which of the following statements regarding this code is correct?

   (A) The code has compilation errors. (B) The code could cause runtime errors.

   (C) The code has no errors.

5. (1 mark) Consider the following Java statement

```
String s = new String("hello");
```

   What value is stored in `s`?

   (A) `"hello"` (B) The object `String("hello")` (C) The address of object `String("hello")`

6. (3 marks) Consider the following code fragment

```
int len = 3, sum = 0, i = 3;
int[] arr = new int[len];
try {
    while (i >= 0) {
         i = i - 1;
         arr[i] = i;
         sum = sum + arr[i];
    }
}
catch (ArrayIndexOutOfBoundsException e) {sum = sum + 1;}
catch (NullPointerException e) {sum = sum - 1;}
catch (Exception e) {sum = 2 * sum;}
```

   arr: int [3].

   i=2 i=1 i=0

   arr[2] = 2. arr[1] = 1 arr[0] = 0

   sum = 2. 3 13

   What value does `sum` have at the end of the execution of the above code?

   (A) 2 (B) 3 (C) 4 (D) 6 (E) 12

7. (2 marks) Consider the following code fragment

```
int[] a = new int[2]; int[] b = new int[2];
for (int i = 0; i < 2; ++i) { a[i] = b[i] = i;}
if (a == b) System.out.println("equal"); // Line 4
else System.out.println("different");
```

   What is the output produced by the above code fragment?

   (A) `"equal"` (B) `"different"` (C) Nothing. Line 4 of the code produces a compilation error.

8. (3 marks) Consider the following Java interface for the queue ADT

```
public interface QueueADT<T> {
   public void enqueue(T element);
   public T dequeue();
}
```

and the following Java code fragment

```
QueueADT<String> s = new QueueADT<String>();
QueueADT<String> q = new QueueADT<String>();
if (s.equals(q)) System.out.println("equal");
else System.out.println("different");
```

What is printed by this above code?

(A) `"equal"` (B) `"different"` (C) Nothing. The above code has compilation errors.

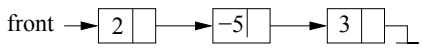
9. (3 marks) Consider the following code fragment

```
public class MyClass {
    private static int value;
    public MyClass (int v) {
         value = v;
    }
    public static void main(String[] args) {
         value = 1;
         MyClass v1 = new MyClass(2);
         MyClass v2 = new MyClass(4);
         System.out.println(v1.value + "," + v2.value);
    }
}
```

What is printed when the above code is executed?

(A) 1,1 (B) 2,2 (C) 2,4 (D) 4,4

10. (3 marks) Consider the following singly linked list, where each node stores an integer value.



The following code fragment is executed on the above list. Method `getValue()` returns the value stored in a node.

```
LinearNode<Integer> p = front;
int[] arr = new int[3];
int count = 3;
try {
    while (count > 0) {
         if (p.getValue() > 0) {
              arr[count-1] = p.getValue();
              count = count-1;
         }
         p = p.getNext();
    }
    System.out.println("Finish");
}
catch (NullPointerException e) {System.out.println("Null pointer");}
catch (IndexOutOfBoundsException e) {System.out.println("Invalid index");}
catch (Exception e) {System.out.println("Error");}
```

What is printed by the above code?

(A) `"Finish"` (B) `"Null pointer"` (C) `"Invalid index"` (D) `"Error"`

(E) `"Null pointer"` and `"Invalid index"` (F) `"Finish"` and `"Invalid index"`

11. (2 marks) Consider the following two Java classes.

```
public class A {
    public void m() {System.out.println("Class A");}
    public A() { }
}
public class B extends A {
    public void m() {System.out.println("Class B");}
    public B() { }
}
```

Consider now the following code fragment

```
A var1 = new B(); // Line 1
B var2 = new A(); // Line 2
```

Which line(s) cause compilation errors?

(A) Line 1 (B) Line 2 (C) Lines 1 and 2 (D) None

12. (4 marks) Consider the two classes from the previous question and the following class

```
public class C extends B {
    public void m() {System.out.println("Class C");}
    public void foo() {System.out.println("Method foo");}
    public C() { }
}
```

Consider the following code fragment

```
C var3 = (C) new A();
var3.foo();
```

What does this code fragment print when it is executed?

(A) `"Method foo"` (B) Nothing. The code has compilation errors.
(C) Nothing. The code produces a runtime error.

13. (3 marks) Consider the same 3 classes from the above two questions and the following code fragment

```
int[] arr = new int[3];
arr[0] = 10;
A var4;
if (arr.length == 3) var4 = new C();
else var4 = new B();
var4.m();
```

What does this fragment print when it is executed?

(A) `"Class A"` (B) `"Class B"` (C) `"Class C"`
(D) Nothing. The code has compilation errors. (E) Nothing. The code produces a runtime error.

14. (2 marks) What is the output produced by the following program?

```
public class C {
    public static String s = "hey";
    public static void changeString(String s) {
        s = "hi";
    }
    public static void main (String[] args) {
        String s = "hello";
        changeString(s);
        System.out.println(s);
    }
}
```

(A) `"hey"` (B) `"hi"` (C) `"hello"`

15. (2 marks) Consider the following code fragment

```
public class A {
    public int i = 1;
    private j = 2;
    public static void main (String[] args) {
        //Switch i and j
        int k = i; // Line 1
        i = j;     // Line 2
        j = k;     // Line 3
    }
}
```

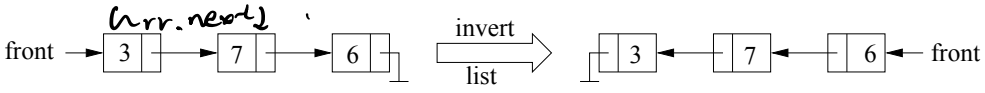
Which line(s) have compilation errors?

(A) Only line 1 (B) Only lines 2 and 3 (C) Lines 1, 2, and 3 (D) None

16. (4 marks) Consider the following code fragment

```
curr = front;
next = curr.getNext();
prev = null;
while (curr != null) {
        (*)
}
front = prev;
```

Which code must be added in the part marked (*) so the above code correctly inverts a non-empty singly linked list? See the figure to understand what "invert" means.



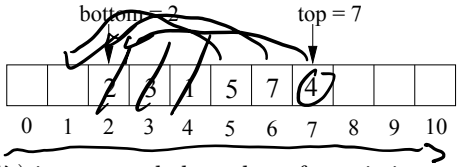
(A) `next.setNext(prev); prev = curr; curr = next; if (next != null) next = next.getNext();`
(B) `curr.setNext(prev); prev = curr; curr = next; if (next != null) next = next.getNext();`
(C) `next.setNext(curr); prev = curr; curr = next; if (next != null) next = next.getNext();`
(D) `prev = curr; curr = next; curr.setNext(prev); if (next != null) next = next.getNext();`

17. (1 mark) Does the code fragment from the previous question also work correctly if the list is empty?

(A) Yes (B) No

18. (3 marks) A *double stack* is a variation of a stack that allows access to the top and to the bottom of the stack. A double stack ADT provides operations `pushTop(T x)` that adds `x` to the top of the stack, `pushBottom(T x)` that adds `x` to the bottom of the stack, `popTop()` that removes the element at the top of the stack, and `popBottom()` that removes the element at the bottom of the stack.

Consider a double stack `ds` implemented using an array as shown in the following figure.



When `pushTop(T x)` (`popTop()`) is executed the value of `top` is increased (decreased); similarly when `pushBottom(T x)` (`popBottom()`) is executed the value of `bottom` is decreased (increased). Consider the following code fragment.

```
for (int i = 0; i < 3; ++i) ds.pushTop(ds.popBottom());
for (int i = 0; i < 2; ++i) x = ds.popTop();
ds.pushBottom(x);
```

After this code fragment is executed on the double stack `ds` represented by the array in the above figure which are the final values for `top` and `bottom`?

(A) top=8, bottom=4 (B) top=9, bottom=5 (C) top=7, bottom=3 (D) top=10, bottom=3

Part II. Written Answers

19. (6 marks) Consider a stack implemented using a singly linked list where `top` and `count` are instance variables pointing to the first node of the list and giving the number of data items in the stack, respectively. The following Java code implements the push operation.

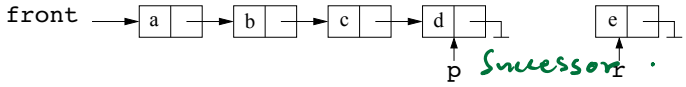
```
private void push (T newValue) {
    LinearNode<T> newNode = new LinearNode<T>(newValue);
    top = newNode;
    newNode.setNext(top.getNext());
    ++count;
}
```

- Is this method correct (circle one)? Yes (No)
- If the method is wrong, give an example showing that the code does not produce the correct result.

  Draw initial stack in your example (show `top` and `count`):

  Draw the stack after performing `push(x)` (show `top` and `count`):

20. (4 marks) Consider the following linked list, and node `r`



Consider the following code intended to add node `r` to the end of the list.

```
successor = p.getNext();
successor = r;
```

Draw the linked list resulting after the above code is executed and indicate whether the code is correct or not.

X

21. (6 marks) Consider the following code.

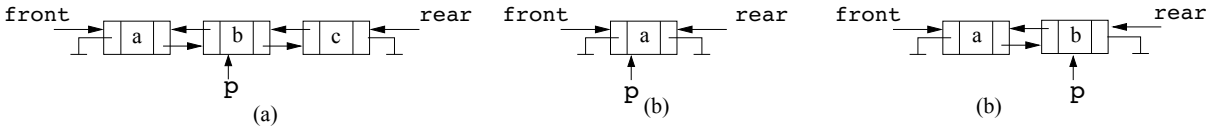
```
public class C {
    private int i;
    public C() {i = 0;}
    private void method1 (int i) throws Exception2 {
        try {
            if (i == 0) throw new Exception2();
            method2 (i);
            this.i = 7;
        } catch (Exception1 e) {
            this.i = 5;
            System.out.println("Exception 1 caught in method1");
        }
    }
    private void method2 (int i) throws Exception1 {
        try {
            if (i > 0) throw new Exception1();
            else throw new Exception2();
        }
        catch (Exception2 e) {System.out.println("Exception 2 caught in method 2");}
    }
    public static void main (String[] args) {
        C objectC = new C();
        try {
            objectC.method1(4);
        }
        catch (Exception2 e) {System.out.println("Exception 2 caught in main");}
        System.out.println("i = " + objectC.i);
    }
}
```

`Exception1` and `Exception2` are not parent/child classes of each other. Write all the output produced when this program is executed.

22. The following code is intended to remove a node `p` from a doubly linked list. Assume that we know that p **is** in the list, so the list is not empty.

```
prev = p.getPrevious();
succ = p.getNext();
if (p == front) {
    front = front.getNext();
    if (front == null) rear = null;
    else front.setPrevious(null);
}
else prev.setNext(succ);
if (p == rear) {
    rear = rear.getPrevious();
    rear.setNext(null);
}
else succ.setPrevious(prev);
```

Consider the following three doubly linked lists



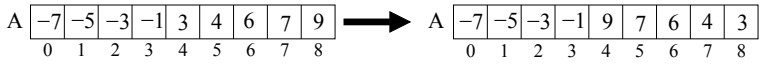
(4 marks) If the above code can be executed on list (a) of the figure without crashing, draw the doubly linked list (including `front` and `rear`) resulting after it is executed; otherwise explain why the code would crash and which statement would throw an exception.

(4 marks) If the above code can be executed on list (b) of the figure without crashing, draw the doubly linked list (including `front` and `rear`) resulting after it is executed; otherwise explain why the code would crash and which statement would throw an exception.

(4 marks) If the above code can be executed on list (b) of the figure without crashing, draw the doubly linked list (including `front` and `rear`) resulting after it is executed; otherwise explain why the code would crash and which statement would throw an exception.

For the following two questions write algorithms in Java or in **detailed** Java-like pseudocode like the one used in the lecture notes.

23. (16 marks) Consider an array `A` storing `n` > 1 **different** integer values sorted in increasing order. At least one value in `A` is negative and at least one of them is positive. Write an algorithm `invertPositive(A,n)` that changes the positions of the **positive** values stored in `A` so the positive values appear in decreasing order. So, for example, if the array `A` is as in the figure on the left, the modified array must be as in the figure on the right.



You can either use a stack or a queue in your solution, **but not both**; you need to decide which of these auxiliary data structures you need to use to answer this question. You **cannot** use any other additional data structures.

The **only** operations that you can perform on the stack are `push`, `pop`, `peek`, and `isEmpty`; you can write `auxStack = new stack` to create an empty stack. If you decide to use a queue the only operations that you can perform on the queue are `enqueue`, `dequeue`, `first`, and `isEmpty`; you can write `auxQueue = new queue` to create an empty queue. You **CANNOT** assume that the stack or the queue are implemented using an array, singly linked list, doubly linked list, or other data structure. The only way to manipulate the stack or the queue is through the use of the above operations.

To manipulate the array you **must** use detailed pseudocode. For example, you can write `A[i] = x` or `s.push(A[i])`, but you **cannot** write statements like "add `x` to the array `A`", or "remove element `x` from `A`".

24. (16 marks) Given a stack `s` storing `n` integer values and an integer value `x`, write an algorithm `inStack(x)` that returns `true` if `x` is in the stack and it returns `false` otherwise. After the execution of the algorithm `s` must be exactly as it was before the algorithm was executed, i.e. it must have the same values and in the same positions.

    Your algorithm can use one additional stack as auxiliary data structure. You **cannot** use any other additional data structures. Write `auxStack = new stack` to create an empty stack, and use methods `push`, `pop`, and `isEmpty` to manipulate a stack. You **CANNOT** assume that the stack is implemented using an array, singly linked list, doubly linked list, or other data structure. The only way to manipulate the stacks is through the use of the above operations.

    Assume that `s` is an instance variable.

    Hint. Move the values in `s` to the auxiliary stack so you can compare them with `x`. Remember that before the algorithm terminates all values must be put back in `s`.

Western University
Department of Computer Science

CS1027b Foundations of Computer Science II
Midterm Exam

March 9, 2019

PART I	
PART II	
19	
20	
21	
22	
23	
24	
Total	

Last Name: _____

First Name: _____

Student Number: _____

Section Number (1- Kontogiannis 2 - Solis-Oba): _____

Instructions

- Fill in your name, student number, and section.
- The exam is 2 hours long and it has a total of 100 marks.
- The exam has 10 pages and 24 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.