Unit 6 – ICS4U0 – ADTs

Sample Test – December 5, 2022

Names: _____

Total	%	Knowledge	Communication	Thinking	Application
(100)		(30)	(22)	(24)	(23)

Knowledge

1. Identify the object method type in each method call by writing in the appropriate first letter. /8

(C)onstructor, (M)utator, (A)ccessor, (F)acilitator

	(a) int num = (int) Math.**random**()*5;
	(b) FileOutputStream out = **openFileOutput**("txt.txt", Activity.MODE_PRIVATE);
	(c) EditText et = **new EditText**();
	(d) String words = et.**getText**();
	(e) textArea.**setText**("Hello");
	(f) if(question.**equals**(answer))
	(g) if(word.**compareTo**(secondWord)>0)
	(h) while(!s.**isFull**())

2. Circle the stack operations in this column.

```
dequeue
enqueue
 insert
isEmpty
  peek
   pop
  push
```

3. Circle the queue operations in this column. /4

```
dequeue
enqueue
 insert
isEmpty
  peek
   pop
  push
```

4. Note what is printed out beside each line of output. /8

```
Queue q = new Queue ();
q.enqueue (9);
q.enqueue (8);
System.out.println (q.dequeue ());          _____
System.out.println (q.size ());             _____
q.enqueue (7);
q.enqueue (6);
System.out.println (q.dequeue ());          _____
System.out.println (q.dequeue ());          _____
System.out.println (q.isEmpty ());          _____
```

```
Stack s = new Stack ();
s.push (9);
s.push (8);
System.out.println (s.pop ());              _____
System.out.println (s.size ());             _____
s.push (7);
s.push (6);
System.out.println (s.pop ());              _____
System.out.println (s.pop ());              _____
System.out.println (s.isEmpty ());          _____
```

5. What does each acronym stand for? /4

O	A	L	F
O	D	I	I
P	T	F	F
		O	O

6. What 11 methods signatures would be needed by the coin class? Fill in the blanks below. /6
A Coin has 3 instance variables: String name, double value, int year.

Constructors	Mutators
_____ _____( )	_____ _____ _____(_____ ____)
_____ _____(_____ __, ____ __, ____ __)	_____ _____ _____(_____ ____)
	_____ _____ _____(_____ ____)
Accessors	
_____ _____ _____()	Others
_____ _____ _____()	_____ _____ _____(_____ ____)
_____ _____ _____()	_____ _____ _____(_____ ____)
_____ _____ _____()	

Communication 

7. Identify the speeds of the following algorithms. /4

(a) pop ..... (b) push .....

(c) dequeue ..... (d) enqueue .....

(e) peek ..... (f) Binary search .....

(g) Quicksort ..... (h) isFull .....

8. In the first column, fill in the terms that match the description. /8

	(a) Variables found inside an object's class.
	(b) A keyword restricting variable access to within the class.
	(c) A template for an object or type. Also contains a java program.
	(d) An object method type that returns values of instance variables.
	(e) An object method type that sets up dynamic memory.
	(f) An object method type that changes the values of instance variables.
	(g) Keeping an object's code self-contained and independent of other code. It relies only on itself.
	(h) Something that shouldn't be public.

9. Choose the best data structure for each description. /5

Stack Queue (a) A FIFO structure.

Stack Queue (b) A LIFO structure.

Stack Queue (c) A pile of books model this data structure well.

Stack Queue (d) The data structure that could model a waiting line.

Stack Queue (e) The data structure used by the browser's back button.

10. What is the trade-off associated with a Stack? /2

. . . . . .

. . . . . .

. . . . . .

. . . . . .

11. What are three characteristics of a class that supports information hiding? Explain each briefly. /3

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Thinking

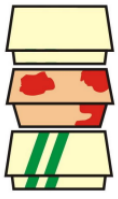
12. Neighbours share a long, narrow, driveway. Cars parked in the driveway leave by backing out. A schedule makes sure that nobody is blocked in when they need to leave. On each day, departing cars leave before any cars enter. Before Monday, no cars are in the driveway. The table shows how the driveway is shared. /2

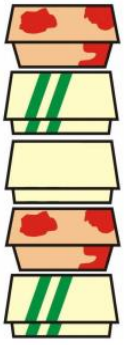
The driveway at the end of Monday is shown below:

Day	# Leaving	# Entering	Owners in enter order
Mon	0	2	Ariadne, Bob
Tues	1	3	Kate, Ben, Roy
Wed	2	1	Daisy
Thurs	0	2	Finn, Rose
Fri	3	1	Vincent

Whose cars will be parked at the end of Friday? (circle)
(A) Bob, Vincent, Daisy (B) Vincent, Ariadne, Rose
(C) Ariadne, Kate, Vincent (D) Ariadne, Daisy, Vincent

13. Alan and Turing are working at a burger restaurant. Alan cooks burgers one at a time. After cooking a burger, he places it into one of three different boxes: one with stripes, one with a pattern and one plain box. If he has cooked three burgers, he would have a stack as shown in diagram A. If he cooked two more burgers, he would have a stack as shown in diagram B. /2


A


B


C

As Alan cooks a burger, he places that box on the top of the stack of not yet sold burgers and continues to cycle through the three different boxes (stripe, pattern, plain, stripe, pattern, plain, ...) to place the burger. Turing sells the burgers one at a time and always takes the uppermost box from the stack. Alan is cooking faster than Turing can sell the burgers. After some time, Turing has sold some burgers and Alan has cooked more burgers.

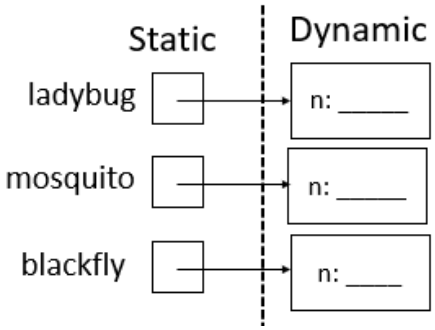
Suppose the stack of unsold burgers looks as shown in diagram C. What is the fewest number of burgers sold by Turing? (circle)

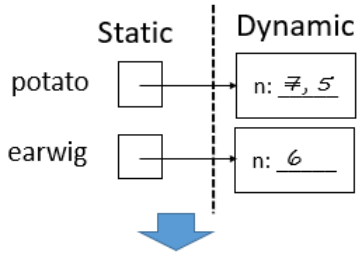
(a) 4
(b) 5
(c) 6
(d) 7

14. Trace this class: fill in both the output (on the lines) and the object diagram. /20

```
public class bug {
   private int n;
   public bug () {
      n = 6;
   }
   public bug (int r) {
      n = r;
   }
   public int buzz () {
      return n + 2;
   }
public int fly () {
   return n;
}
public void setB(int r){
   n = r;
}
public void minusTwo () {
   n-=2;
}
public String toString () {
   return ""+ n;
public boolean equals (bug b) {
   return b.buzz () == (n + 2);
}
public int compareTo (bug b) {
   if ((n + 2) == b.buzz ())
      return 0;
   else if ((n + 2) > b.buzz ())
      return 1;
   else
      return -1;
}}
```

```
bug ladybug = new bug ();
System.out.println(ladybug.toString ());                .....
bug mosquito = new bug (6);
System.out.println(mosquito.toString ());               .....
System.out.println(ladybug.buzz ());                    .....
bug blackfly = new bug (10);
System.out.println(blackfly.toString ());               .....
System.out.println(mosquito.equals (ladybug));          .....
System.out.println(blackfly.compareTo (mosquito));      .....
System.out.println(mosquito.compareTo (blackfly));      .....
System.out.println(mosquito.fly());                     .....
mosquito.minusTwo();
System.out.println(mosquito.fly());                     .....
blackfly.buzz();
System.out.println(blackfly.fly());                     .....
```





15. Using the above bug class and the above object diagram, write the code to produce the output shown.

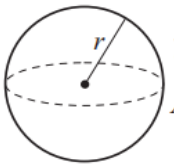
Printed on the screen	***Code***
(declare the two objects)	
7	
8	
5	
6	

Application

16. Fill in the Sphere class, then edit the Queue class to make a Queue of Spheres. /16

```
public class Sphere {
  private int radius;
  public Sphere() {


  }
  public Sphere(int r) {


  }
  public void setRadius(int r) {


  }
  public int getRadius() {


  }
  public String toString() {









  }
  public double surfaceArea() {
    return 4*Math.PI*radius*radius;
  }
  public double volume() {









  }
  public boolean equals(Sphere s) {
  //return true if equal, false otherwise









  }
  public int compareTo(Sphere s) {
  //returns 1 if radius is biggest
    if(radius>s.getRadius())
      return ____;
    //returns 0 if equal
    else if (radius==s.getRadius())
      return ____;
    //return -1 otherwise
    else
      return ____;
  }
}
```

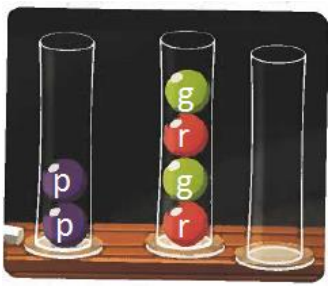
```
public class Queue {
  private Object data[] = new Object [50];
  int count;
  int head;

  public Queue () {
    count = 0;
    head = 0;
  }

  public void enqueue (Object value) {
    int tail = (head + count) % data.length;
    data [tail] = value;
    count++;
  }

  public Object dequeue () {
    Object temp = data [head];
    count--;
    head = (head + 1) % data.length;
    return temp;
  }

  public Object peek () {
    return data [head];
  }

  public int size () {
    return count;
  }

  public boolean isEmpty () {
    return (count == 0);
  }
}
```



Call your new Queue in the MainActivity class.

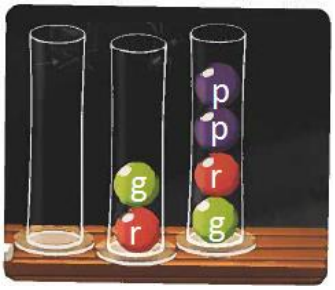
```
public void makeQueue(View view) {
    Queue q = new _____();
    q.enqueue(new Sphere(____));
    Sphere m = new Sphere(____);
    q.enqueue(___);
    q.dequeue();
}
```

17. (a) Fill in the code to make these 3 Stacks.



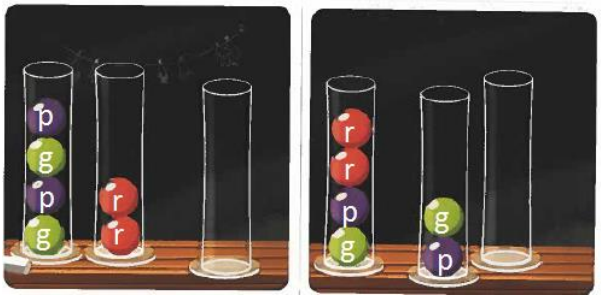

```
s1.push("____");
s1.push("____");

s2.push("____");
s2.push("____");
s2.push("____");
s2.push("____");
```

(b) Fill in the code. Change the previous to this picture.




```
s3.push(___.pop());
s3.push(___.pop());

s3.push(___.pop());
s3.push(___.pop());
```

(c) What is the smallest number of steps you can take to transfer the stack in the first picture to the second?

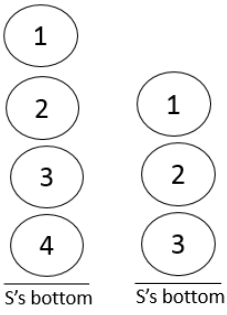


Smallest number of steps?. . . . . .

/3

18. Remove and print out the bottom element of the Stack named 's'. After you are done, the rest of the Stack should be in its original order. The picture shows **an example** of this process.

/5



Prints: 4