

Read each question carefully. When writing code, note that not all questions require a complete class, or even a complete method.

1. What is the significance of the **main** method? Why must it be **static**? (2 pts)

The main method is the start of the program. Non-static methods are called on objects – main must be static because you cannot build any objects until the program starts, but you would need at least one such object to be built before you could call the main method.

2. What is an immutable class? (2 pts)

A class with no mutator methods is immutable.

3. Consider the Java code below & determine what the final output is (2 pts)

```
int x=0;
for(int a=0; a<4; a++)
{
    for(int b=0; b<a; b++)
    {
        x++;
    }
}

System.out.println("x: " + x);
```

| | |
|--------------------------------------|-------|
| x = 0 | |
| a = 0, b = 0 to 0 | x = 0 |
| a = 1, b = 0 to 1 x++ | x = 1 |
| a = 2, b = 0 to 2 x++, x++ | x = 3 |
| a = 3, b = 0 to 3 x++, x++, x++ | x = 6 |
| a = 4, done | |

4. Write some Java code that gets an integer from the user, builds an **array** of that size and then takes that many **Strings** as input from the user & stores them in array using a **for** loop. You do not need to write an entire class or even an entire method, just a short section of code. (6 pts)

```
import java.util.Scanner;
...
int k = in.nextInt();
String[] array = new String[k];
for(int x=0; x<k; x++)
{
    array[x] = in.next();
}
```

5. Write a method that takes an **ArrayList** of **Strings** and returns the longest one. Use a **for each** loop. Include documentation comments, including a documentation comment stating the precondition that the ArrayList must not be empty. (9 pts)

```
/** returns longest String from list
 * (precondition) : list.size() > 0
 * @param list the ArrayList to check
 * @return longest String in the list
 */
public String biggestString(ArrayList<String> list)
{
    String longest = "";
    for(String s : list)
    {
        if( s.length() > longest.length() )
            longest = s;
    }
    return longest;
}
```

technically, this would be better:

```
longest = new String(s);
```

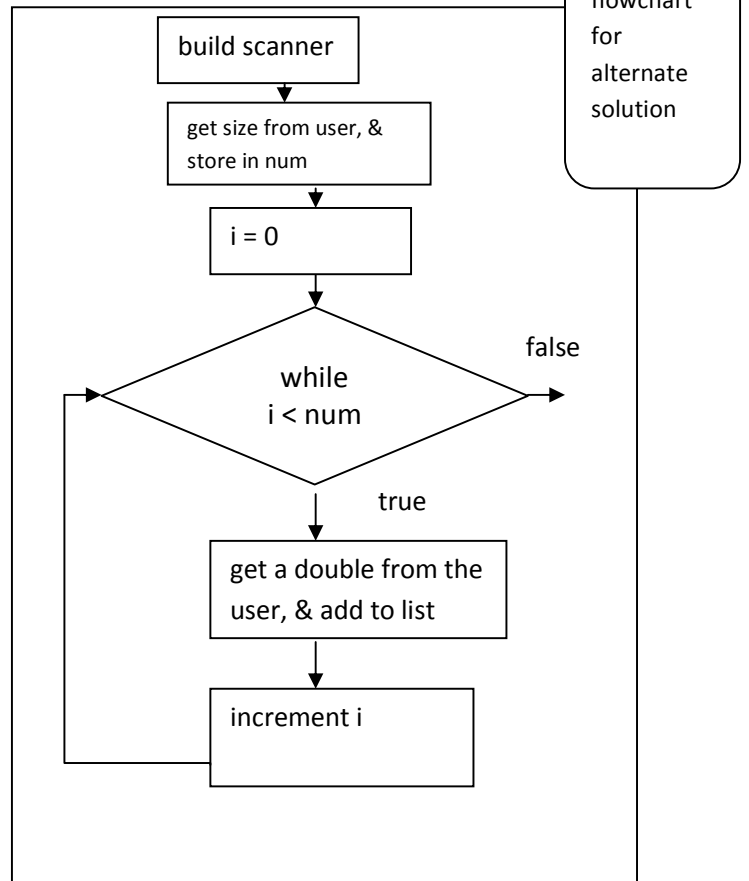
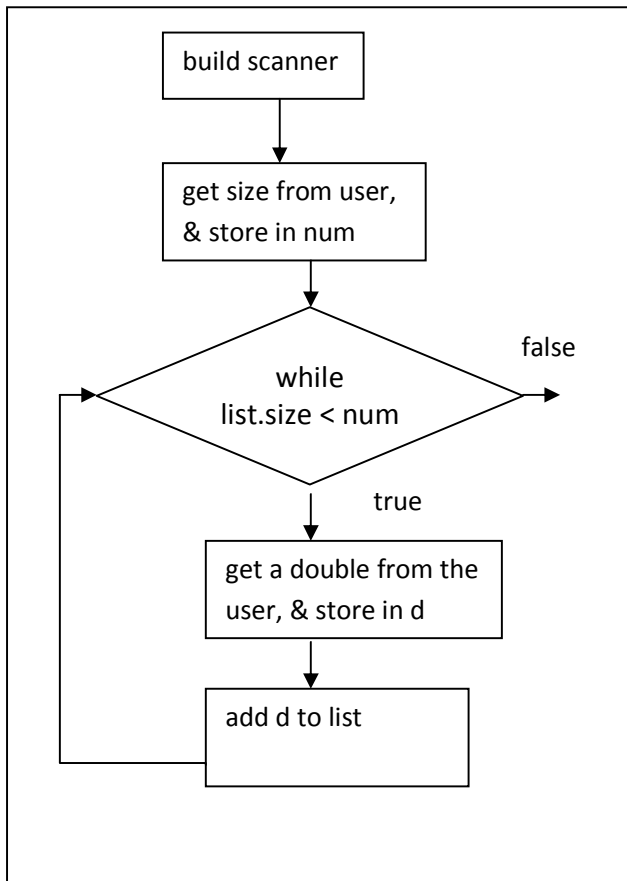
6. Write some Java code that gets an integer from the user, and then takes that many doubles as input from the user & stores them in **ArrayList** using a **while** loop. You do not need to write an entire class or even an entire method, just a short section of code. (6 pts)

```
import java.util.Scanner;
...
Scanner in = new Scanner(System.in);
int num = in.nextInt();
ArrayList<Double> list = new ArrayList<Double>();
while(list.size() < num)
{
    double d = in.nextDouble();
    list.add(d);
}
```

```
import java.util.Scanner;
...
Scanner in = new Scanner(System.in);
int num = in.nextInt();
ArrayList<Double> list = new
ArrayList<Double>();
int i = 0;
while(i < num)
{
    list.add( in.nextDouble() );
    i++;
}
```

alternate solution

7. Draw out a flow chart for the code you wrote in the problem above. (5 pts)



flowchart for alternate solution

8. Write out a single line of code to test if the value y is between the values x and z. (3 pts)

```
System.out.print("enter 3 integers (x, y, z): ");
// get input from user & save it
int x = in.nextInt();
int y = in.nextInt();
int z = in.nextInt();
```

```
// your code goes below here
```

```
if( (x < y && y < z) || (z < y && y < x) )
```

```
// your code goes above here
```

```
{
    System.out.println("y is between x and z");
}
else
{
    System.out.println("y is not between x and z");
}
```

9. Write a method that takes an integer & returns the number of factors it has. x is a factor of y if and only if y can be evenly divided by x.(6 pts)
Include documentation comments (3 pts)

```
/** deteremines the number of factors for a given number
 *
 * @param num the number to check
 * @return the count of the factors
 */
public int numberOfFactors(int num)
{
    int count = 0;
    for(int a = 1; a <= num; a++)
    {
        if((num % a) == 0)
            count++;
    }
    return count;
}
```

10. Write the code to set the value of the variable price based on the variables age & balconySeating (balconySeating is true if the tickets are for the balcony area, otherwise it is false). Use the chart below to determine the prices. (6 pts)

| | Age: | | |
|---------|------|-------|------|
| Area: | 0-6 | 7-59 | 60+ |
| Floor | Free | 5.00 | 4.00 |
| Balcony | free | 10.00 | 9.00 |

```
int age;  
boolean balconySeating;  
double price;  
  
// code to set the values of age & balconySeating  
...  
// your code goes below here
```

```
if(age <= 6)  
{  
    price = 0.0;  
}  
else if(age <= 59)  
{  
    if(balconySeating)  
        price = 10.0;  
    else  
        price = 5.0;  
}  
else  
{  
    if(balconySeating)  
        price = 9.0;  
    else  
        price = 4.0;  
}
```

Extra credit (3 pts): Explain one concept not asked about in this exam from the last 4 chapters.