

Name _____ Period _____

1. Complete the stack/heap diagram for the following code segment. Then, indicate what is printed.

```
public class BankDriver {

    static BankAccount myAccount = new BankAccount(200); //sets balance to 200

    public static void main(String[] args)
    {
        double b[] = new double[5];
        b[3] = 24;
        int x = 12;
        method1(x, b, myAccount);
        BankAccount anotherAccount = new BankAccount(600); //sets balance = 600
        anotherAccount = myAccount;
        System.out.println(x + " " + b[3] + " " + myAccount.balance);
    }

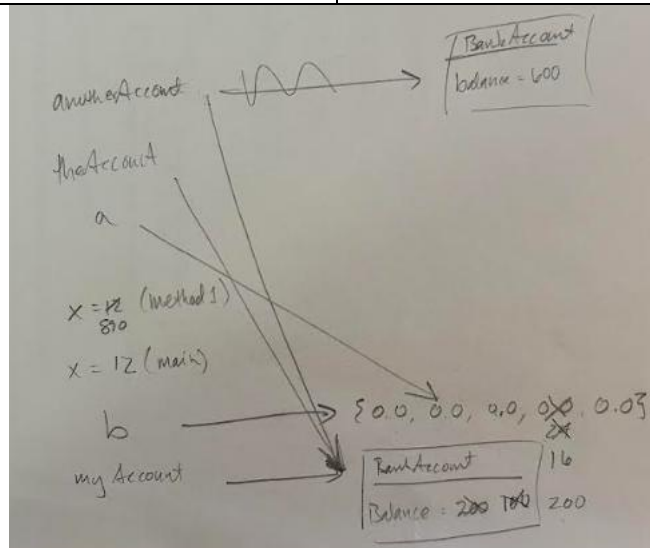
    public static void method1(int x, double a[], BankAccount theAccount){

        x = 890;
        a[3] = 16;
        theAccount.balance = 100;
        myAccount.deposit(100); //adds 100 to the balance
    }

}
```

Stack

Heap



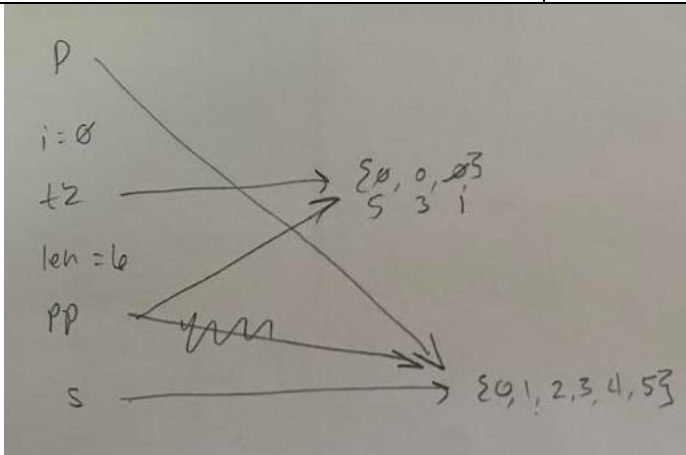
What is printed when the code above is executed?

12 16 200

/4

2. Complete the stack/heap diagram for the following code segment. Then, indicate what is printed.

```
public class Testing3 {  
    public static void main(String args[]){  
        int s[] = {0,1,2,3,4,5};  
  
        for(int g = 0; g < s.length; g++){  
            System.out.print(s[g] + " ");  
  
            System.out.print("\n");  
            testMethod(s);  
  
            int p[] = s;  
  
            for(int g = 0; g < s.length; g++){  
                System.out.print(p[g] + " ");  
            }  
  
            public static void testMethod(int pp[]){  
                int len = pp.length;  
                int t2[] = new int[len/2];  
                int i = 0;  
  
                for(int j=len/2-1; j>=0; j--){  
                    i+=2;  
                    t2[j] = pp[i-1];  
                }  
                for(int k=0; k<t2.length; k++){  
                    System.out.print(t2[k] + " ");  
  
                    System.out.print("\n");  
                    pp = t2;  
                }  
            }  
        }  
    }
```

Stack	Heap
	
<p>What is printed when the code above is executed?</p> <p>0 1 2 3 4 5</p> <p>5 3 1</p> <p>0 1 2 3 4 5</p>	
<div style="border: 1px solid black; width: 50px; height: 30px; display: flex; align-items: center; justify-content: center;">/4</div>	

3. Complete the stack/heap diagram for the following code segment. Then, indicate what is printed.

```
public class Testing
{
    public static void main(String args[])
    {
        int [] prf = {2,3,8,2};
        double d = 4.58;
        Circle myCir = new Circle(11); //set rad = 11
        myCir.rad = 25;
        fg(prf, d, myCir);

        System.out.println(d);
        System.out.println(prf[0]);
        System.out.println(myCir.rad);
    }

    public static void fg(int [] x, double d, Circle c)
    {
        d++;
        x[0] = 15;
        c.rad = 34;
        System.out.println(++d);
    }
}
```

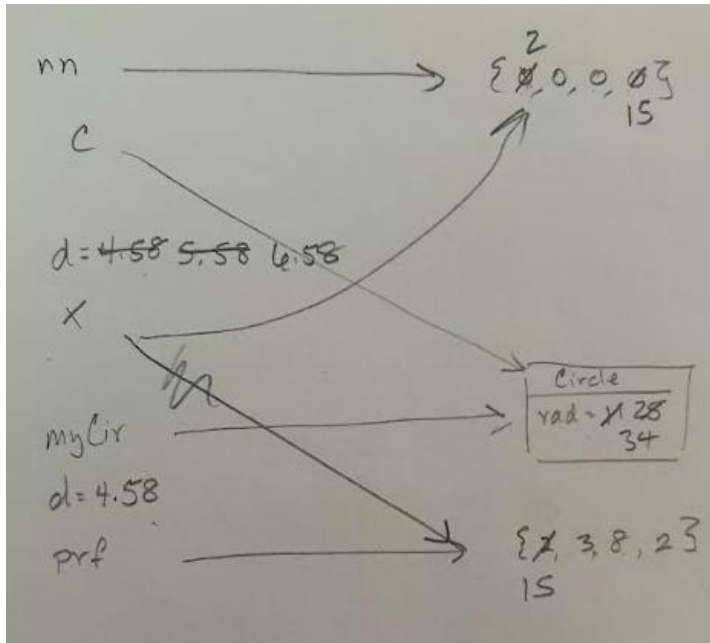
```

    int nn[] = new int[x.length];
    nn[3] = x[0];
    x = nn;
    x[0] = 2;
}
}

```

Stack

Heap



What is printed when the code is executed?

6.58
4.58
15
34

/4

4. This question involves the implementation of a class, called **StringManip**, which is used to perform manipulation on strings.

The class provides the following methods,

- **removeSpaces**, takes a string and returns a new string with the spaces removed. For example, `removeSpaces("hi how are you")`, returns "hihowareyou".
- **reverseString**, which takes a string and returns a new string with the characters in reverse order. For example, `reverseString("ABCDE")` should return "EDCBA".
- **palindromeChecker**, takes a string and determines whether the string is a palindrome and prints a message indicating the result. Examples of the intended behavior of the method are shown in the following table.

Method Call	Printed Message
palindromeChecker("taco cat")	taco cat is a palindrome
palindromeChecker("laid on no dial")	laid on no dial is a palindrome
palindromeChecker("level up")	level up is not a palindrome

The following table illustrates how the `StringManip` class works.

Statements and Expressions	Value Returned (blank if no value)	Comment
<code>StringManip.removeSpaces("laid on no dial");</code>	laidonnodial	Returns a string with the spaces removed
<code>StringManip.reversString("laid on no dial");</code>	laid on no dial	Returns a string in reverse order
<code>StringManip.palindromeChecker("laid on no dial")</code>	laid on no dial is a palindrome	Returns a message indicating whether or not the string is a palindrome
<code>StringManip.palindromeChecker("taco cat")</code>	taco cat is a palindrome	Returns a message indicating whether or not the string is a palindrome
<code>StringManip.palindromeChecker("level up");</code>	level up is not a palindrome	Returns a message indicating whether or not the string is a palindrome

Write the complete `StringManip` class, including the necessary constructors and any required instance variables and methods. Your implementation must meet all specifications and conform to the example.

```
public static String removeSpaces(String phrase){
    String result = "";
    for(int letter = 0; letter < phrase.length(); letter++){
        if(!phrase.substring(letter, letter+1).equals(" ")){
            result += phrase.substring(letter, letter+1);
        }
    }
    return result;
}

public static String reverseString(String phrase){
    String result = "";
    for(int letter = phrase.length() - 1; letter >=0; letter--){
        result += phrase.substring(letter, letter + 1);
    }
    return result;
}

public static String palindromeChecker(String phrase){
    String result = removeSpaces(phrase);
    String reverse = reverseString(result);
    if(result.equals(reverse)){
        return phrase + " is a palindrome";
    }
    return phrase + " is not a palindrome";
}
```

5. The following class represents a **customer**. The variable **name** represents the name of the customer, and the variable **currAccNum** represents the customer's account number. Each time a **Customer** object is created, the static variable **nextAccNum** is used to assign the customer's account number.

```
public class Customer
{
    private static int nextAccNum = 1;
    private String name;
    private int currAccNum;

    public Customer(String n) {
        name = n;
        currAccNum = nextAccNum;
        nextAccNum++;
    }
}
```

- (a) Write a method for the **Customer** class that that will return a string representing a bill notice when passed a double value representing an amount due.

For example, if the customer has name "Jeremiah", has account number 3, and has amount due 50.50, the method should return a string in the following format.

Jeremiah, account number 3, please pay \$50.50

Write the method below. Your implementation must conform to the example above.

```
public String generateBill(double amount)
{
    return name + ", account number " + currAccNum +
           ", please pay $" + amount;
}
```

/2

(b) Write a method for the `Customer` class that returns the value of the next account number that will be assigned.

```
public static int getNextAccNum()
{
    return nextAccNum;
}
```

/2

(c) A student has written the following method to be included in the `Customer` class. The method is intended to update the name of a customer but does not work as intended.

```
public void updateName(String name)
{
    name = name;
}
```

Write a correct implementation of the `updateName` method that avoids the error in the student's implementation.

```
public void updateName(String newName)
{
    name = newName;
}
```

Or

```
public void updateName(String name)
{
    this.name = name;
}
```

/1