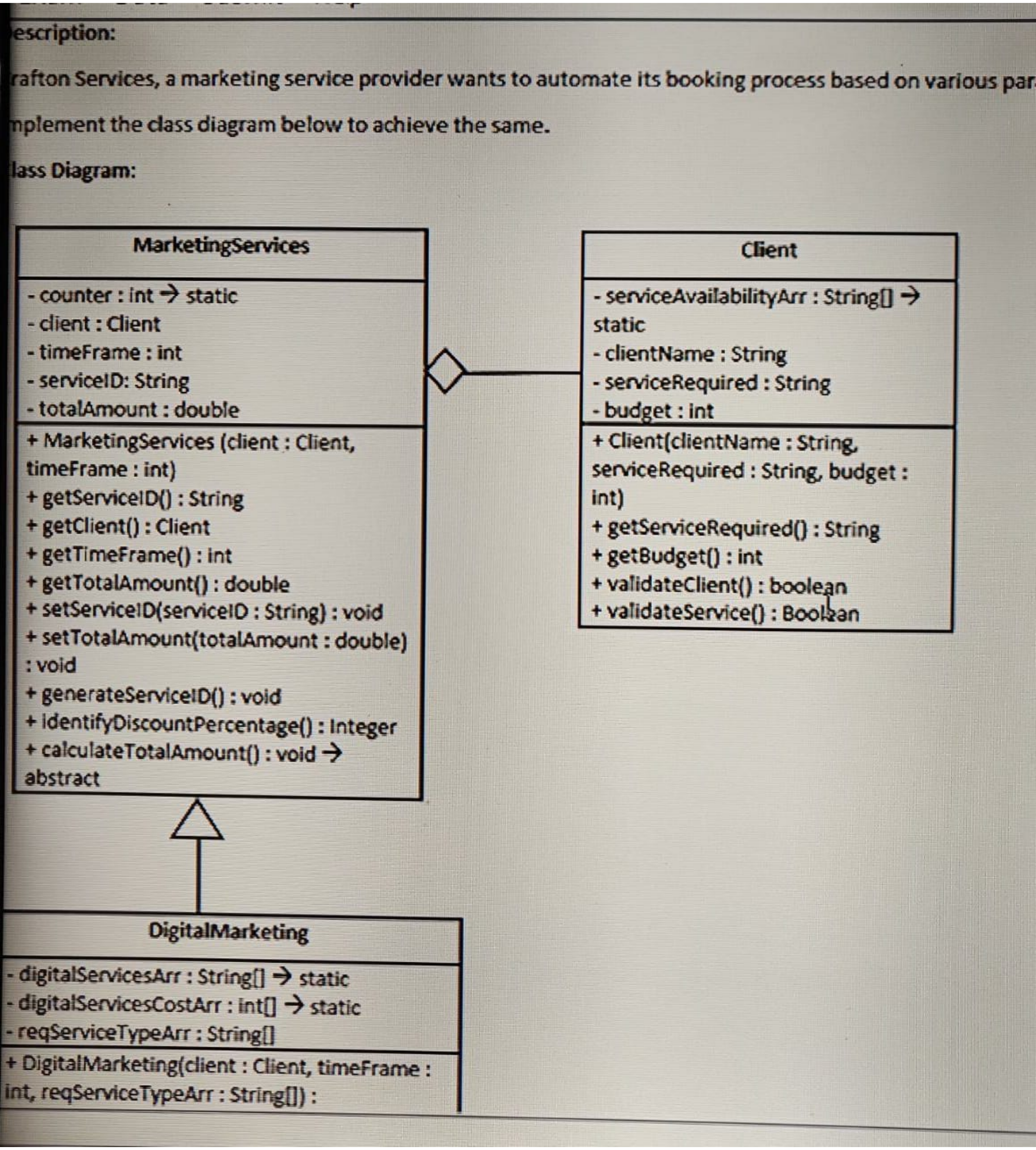


escription:

rafton Services, a marketing service provider wants to automate its booking process based on various par

nplement the class diagram below to achieve the same.

lass Diagram:

MarketingServices

- counter : int → static
- client : Client
- timeFrame : int
- serviceID: String
- totalAmount : double

+ MarketingServices (client : Client, timeFrame : int)
+ getServiceID() : String
+ getClient() : Client
+ getTimeFrame() : int
+ getTotalAmount() : double
+ setServiceID(serviceID : String) : void
+ setTotalAmount(totalAmount : double) : void
+ generateServiceID() : void
+ identifyDiscountPercentage() : Integer
+ calculateTotalAmount() : void → abstract

Client

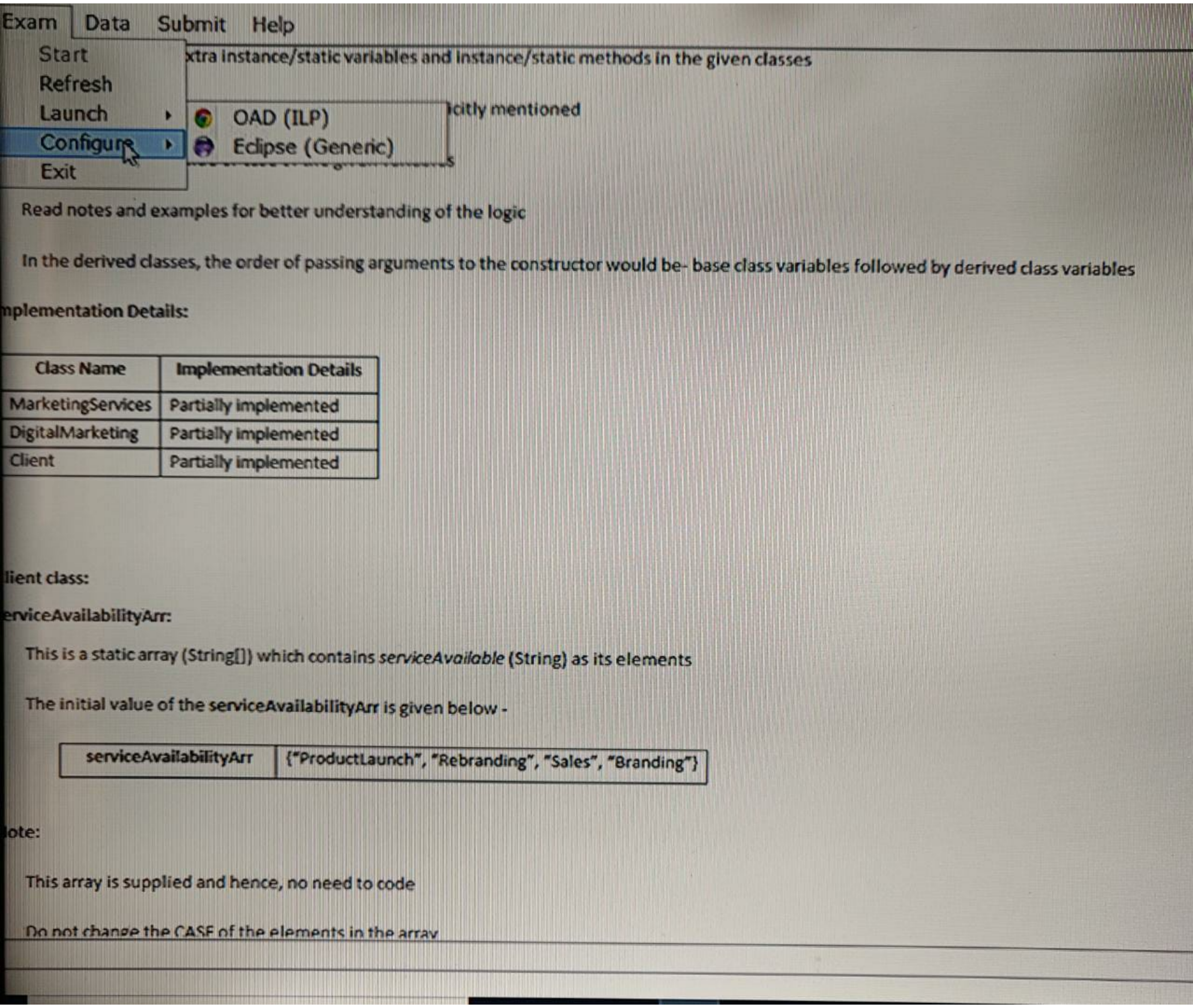
- serviceAvailabilityArr : String[] → static
- clientName : String
- serviceRequired : String
- budget : int

+ Client(clientName : String, serviceRequired : String, budget : int)
+ getServiceRequired() : String
+ getBudget() : int
+ validateClient() : boolean
+ validateService() : Boolean

DigitalMarketing

- digitalServicesArr : String[] → static
- digitalServicesCostArr : int[] → static
- reqServiceTypeArr : String[]

+ DigitalMarketing(client : Client, timeFrame : int, reqServiceTypeArr : String[]) :



xtra instance/static variables and instance/static methods in the given classes

icitly mentioned

s

Read notes and examples for better understanding of the logic

In the derived classes, the order of passing arguments to the constructor would be- base class variables followed by derived class variables

nplementation Details:

Class Name	Implementation Details
MarketingServices	Partially implemented
DigitalMarketing	Partially implemented
Client	Partially implemented

lient class:

erviceAvailabilityArr:

This is a static array (String[]) which contains *serviceAvailable* (String) as its elements

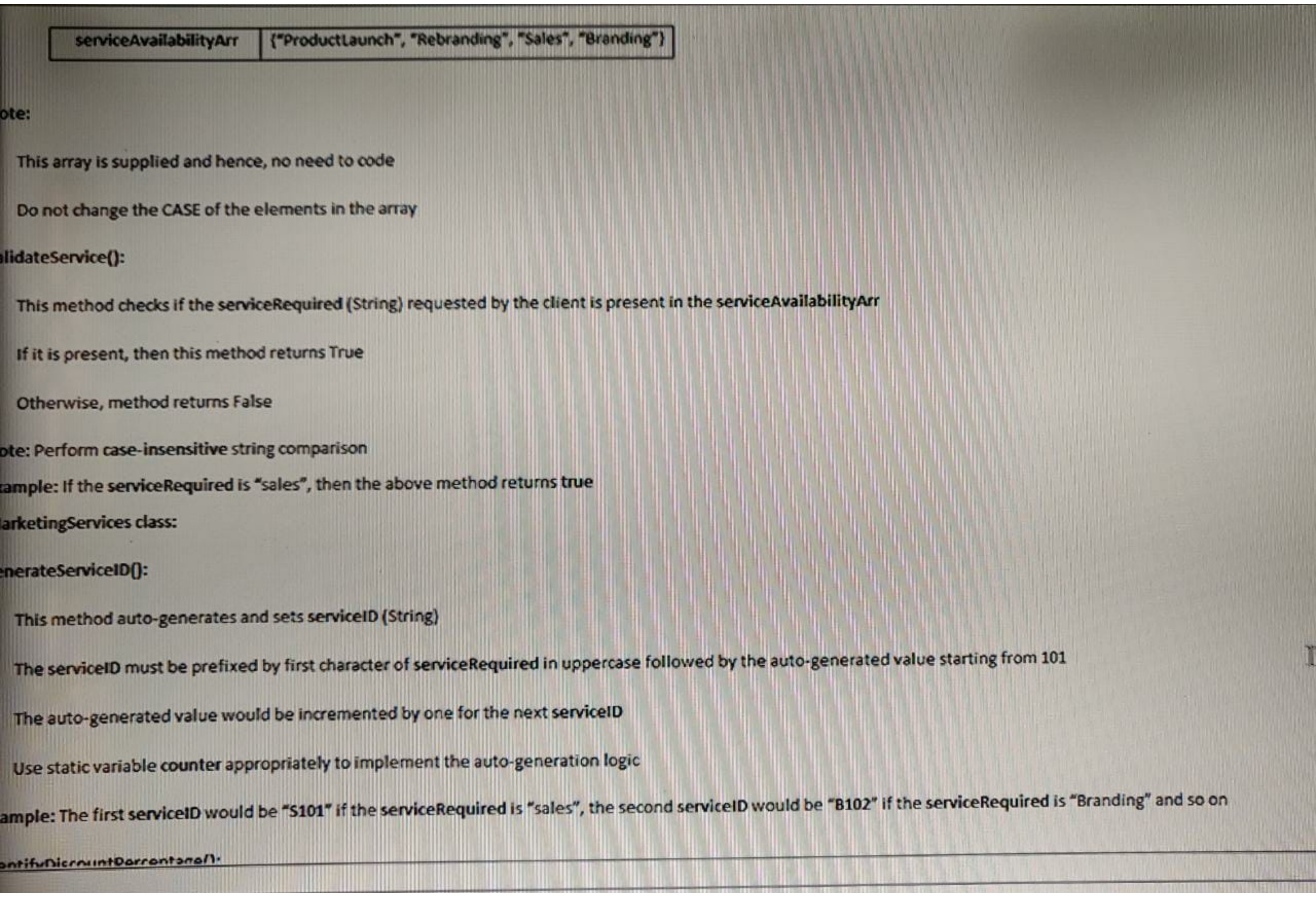
The initial value of the **serviceAvailabilityArr** is given below -

serviceAvailabilityArr	{"ProductLaunch", "Rebranding", "Sales", "Branding"}

ote:

This array is supplied and hence, no need to code

Do not change the CASE of the elements in the array



serviceAvailabilityArr	{"ProductLaunch", "Rebranding", "Sales", "Branding"}

ote:

This array is supplied and hence, no need to code

Do not change the CASE of the elements in the array

alidateService():

This method checks if the **serviceRequired** (String) requested by the client is present in the **serviceAvailabilityArr**

If it is present, then this method returns True

Otherwise, method returns False

ote: Perform **case-insensitive** string comparison

ample: If the **serviceRequired** is "sales", then the above method returns **true**

arketingServices class:

nerateServiceID():

This method auto-generates and sets **serviceID** (String)

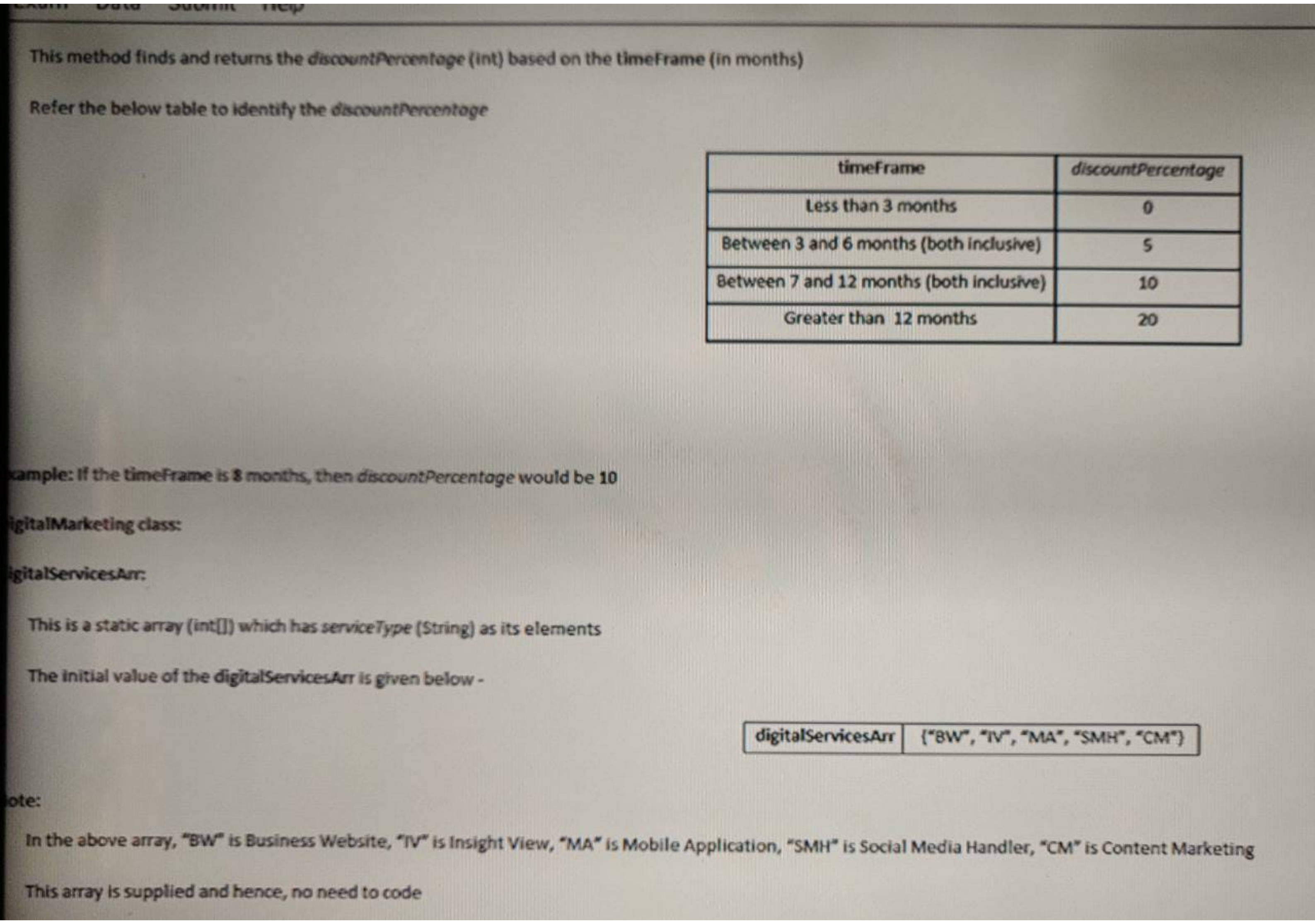
The **serviceID** must be prefixed by first character of **serviceRequired** in uppercase followed by the auto-generated value starting from 101

The auto-generated value would be incremented by one for the next **serviceID**

Use static variable **counter** appropriately to implement the auto-generation logic

ample: The first **serviceID** would be "S101" if the **serviceRequired** is "sales", the second **serviceID** would be "B102" if the **serviceRequired** is "Branding" and so on

entifyDiscountPercentage():



This method finds and returns the *discountPercentage* (int) based on the timeFrame (in months)

Refer the below table to identify the *discountPercentage*

timeFrame	*discountPercentage*
Less than 3 months	0
Between 3 and 6 months (both inclusive)	5
Between 7 and 12 months (both inclusive)	10
Greater than 12 months	20

ample: If the timeFrame is 8 months, then *discountPercentage* would be 10

igitalMarketing class:

igitalServicesArr:

This is a static array (int[]) which has *serviceType* (String) as its elements

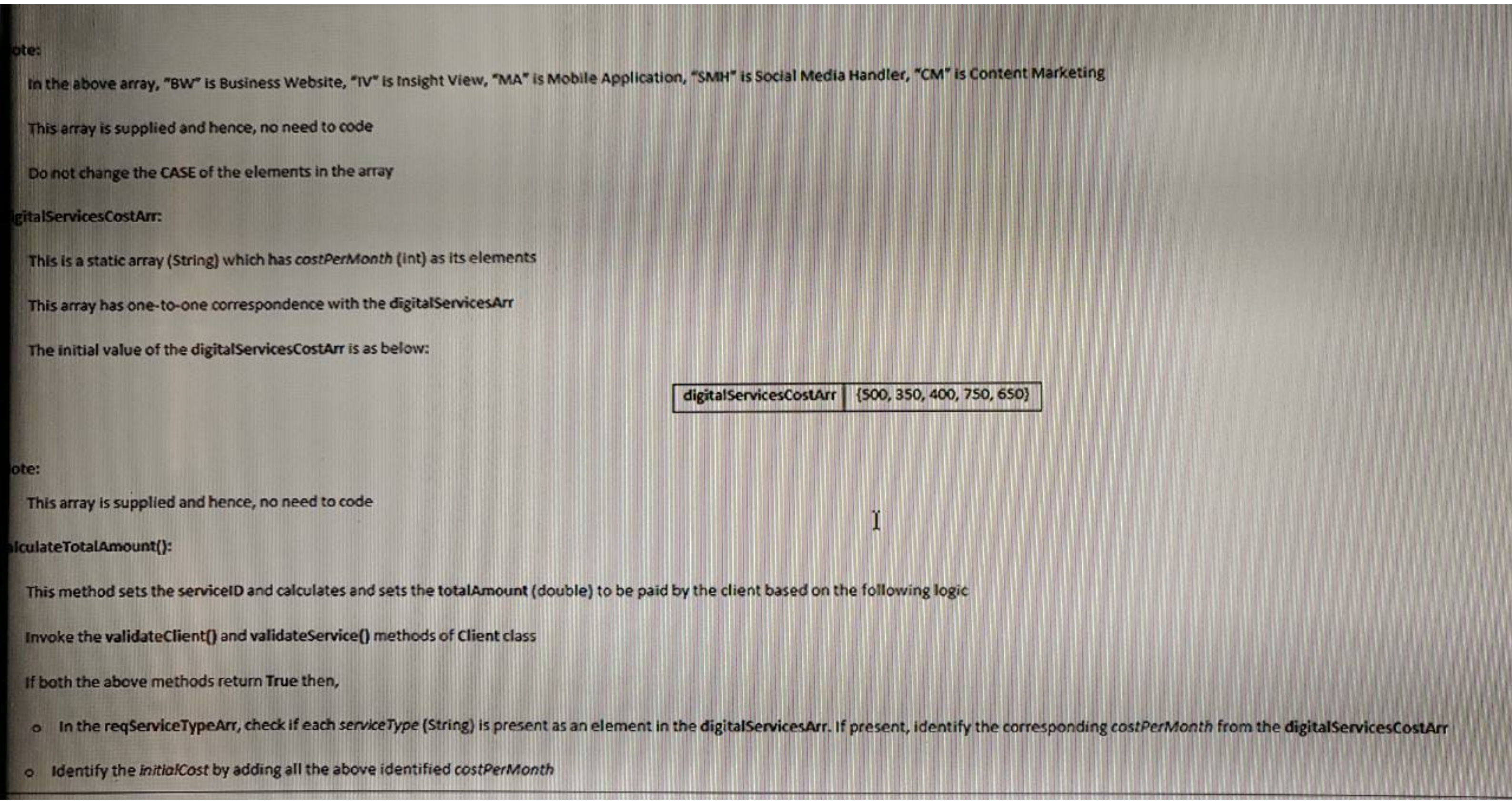
The initial value of the digitalServicesArr is given below -

digitalServicesArr	{"BW", "IV", "MA", "SMH", "CM"}

ote:

In the above array, "BW" is Business Website, "IV" is Insight View, "MA" is Mobile Application, "SMH" is Social Media Handler, "CM" is Content Marketing

This array is supplied and hence, no need to code



ote:

In the above array, "BW" is Business Website, "IV" is Insight View, "MA" is Mobile Application, "SMH" is Social Media Handler, "CM" is Content Marketing

This array is supplied and hence, no need to code

Do not change the CASE of the elements in the array

gitalServicesCostArr:

This is a static array (String) which has *costPerMonth* (int) as its elements

This array has one-to-one correspondence with the **digitalServicesArr**

The initial value of the **digitalServicesCostArr** is as below:

digitalServicesCostArr	{500, 350, 400, 750, 650}

ote:

This array is supplied and hence, no need to code

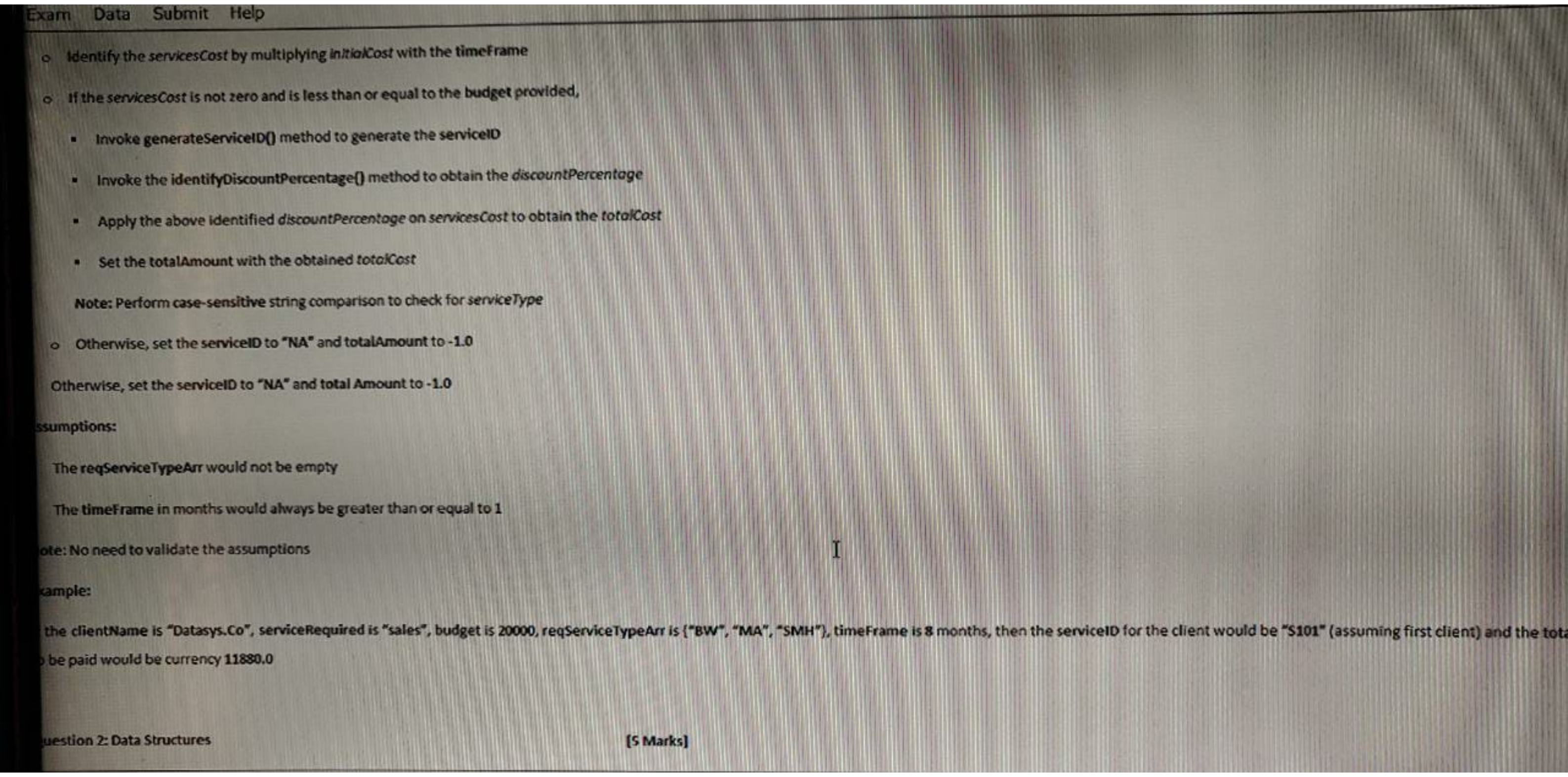
lculateTotalAmount():

This method sets the serviceID and calculates and sets the **totalAmount** (double) to be paid by the client based on the following logic

Invoke the **validateClient()** and **validateService()** methods of **Client** class

If both the above methods return **True** then,

- In the **reqServiceTypeArr**, check if each *serviceType* (String) is present as an element in the **digitalServicesArr**. If present, identify the corresponding *costPerMonth* from the **digitalServicesCostArr**
- Identify the *initialCost* by adding all the above identified *costPerMonth*



- Identify the *servicesCost* by multiplying *initialCost* with the **timeFrame**
- If the *servicesCost* is not zero and is less than or equal to the **budget** provided,
    - Invoke **generateServiceID()** method to generate the **serviceID**
    - Invoke the **identifyDiscountPercentage()** method to obtain the *discountPercentage*
    - Apply the above identified *discountPercentage* on *servicesCost* to obtain the *totalCost*
    - Set the **totalAmount** with the obtained *totalCost*

  Note: Perform **case-sensitive** string comparison to check for *serviceType*
- Otherwise, set the **serviceID** to "NA" and **totalAmount** to **-1.0**

Otherwise, set the **serviceID** to "NA" and **total Amount** to **-1.0**

ssumptions:

The **reqServiceTypeArr** would not be empty

The **timeFrame** in months would always be greater than or equal to **1**

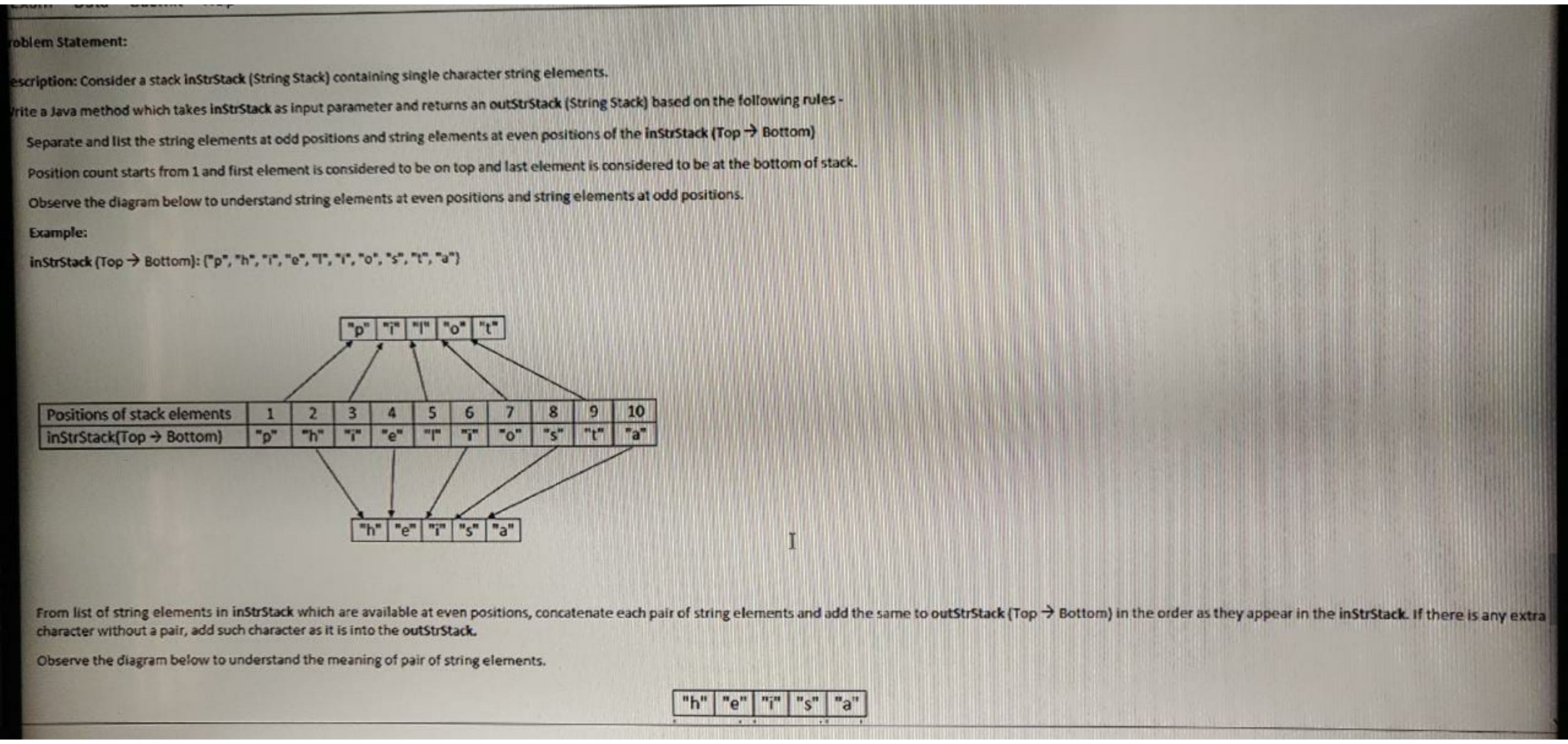
ote: No need to validate the assumptions

ample:

the **clientName** is "Datasys.Co", **serviceRequired** is "sales", **budget** is 20000, **reqServiceTypeArr** is {"BW", "MA", "SMH"}, **timeFrame** is **8** months, then the **serviceID** for the client would be **"S101"** (assuming first client) and the tota

be paid would be currency **11880.0**

uestion 2: Data Structures **[5 Marks]**



roblem Statement:

escription: Consider a stack **inStrStack** (String Stack) containing single character string elements.

rite a Java method which takes **inStrStack** as input parameter and returns an **outStrStack** (String Stack) based on the following rules -

Separate and list the string elements at odd positions and string elements at even positions of the **inStrStack (Top → Bottom)**

Position count starts from 1 and first element is considered to be on top and last element is considered to be at the bottom of stack.

Observe the diagram below to understand string elements at even positions and string elements at odd positions.

Example:

inStrStack (Top → Bottom): {"p", "h", "i", "e", "l", "i", "o", "s", "t", "a"}

"p" "i" "l" "o" "t"

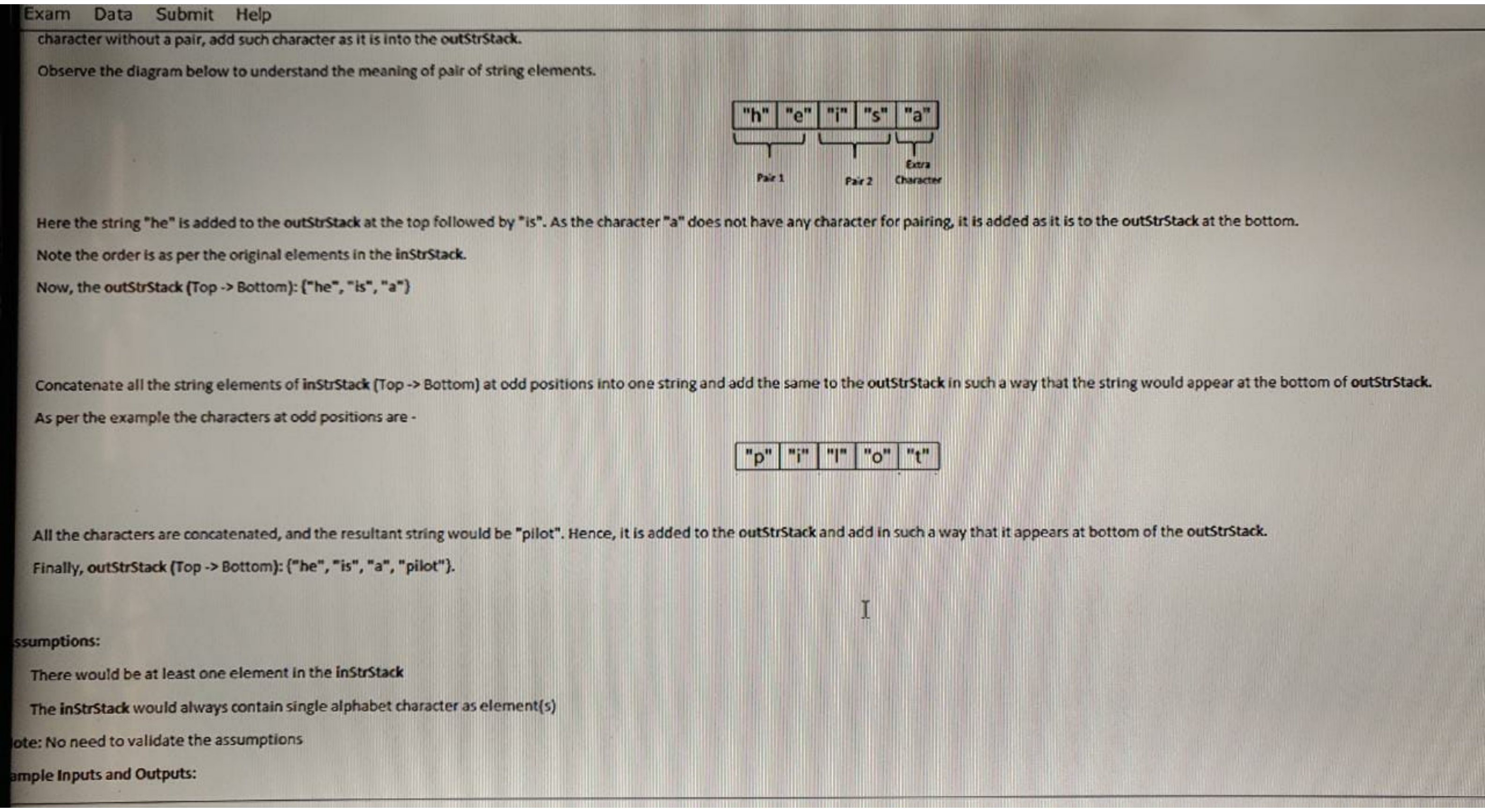
Positions of stack elements	1	2	3	4	5	6	7	8	9	10
inStrStack(Top → Bottom)	"p"	"h"	"i"	"e"	"l"	"i"	"o"	"s"	"t"	"a"

"h" "e" "i" "s" "a"

From list of string elements in **inStrStack** which are available at even positions, concatenate each pair of string elements and add the same to **outStrStack** (Top → Bottom) in the order as they appear in the **inStrStack**. If there is any extra character without a pair, add such character as it is into the **outStrStack**.

Observe the diagram below to understand the meaning of pair of string elements.

"h" "e" "i" "s" "a"



character without a pair, add such character as it is into the **outStrStack**.

Observe the diagram below to understand the meaning of pair of string elements.

"h"	"e"	"i"	"s"	"a"

Here the string "he" is added to the **outStrStack** at the top followed by "is". As the character "a" does not have any character for pairing, it is added as it is to the **outStrStack** at the bottom.

Note the order is as per the original elements in the **inStrStack**.

Now, the **outStrStack** (Top -> Bottom): {"he", "is", "a"}

Concatenate all the string elements of **inStrStack** (Top -> Bottom) at odd positions into one string and add the same to the **outStrStack** in such a way that the string would appear at the bottom of **outStrStack**.

As per the example the characters at odd positions are -

"p"	"i"	"l"	"o"	"t"

All the characters are concatenated, and the resultant string would be "pilot". Hence, it is added to the **outStrStack** and add in such a way that it appears at bottom of the **outStrStack**.

Finally, **outStrStack** (Top -> Bottom): {"he", "is", "a", "pilot"}.

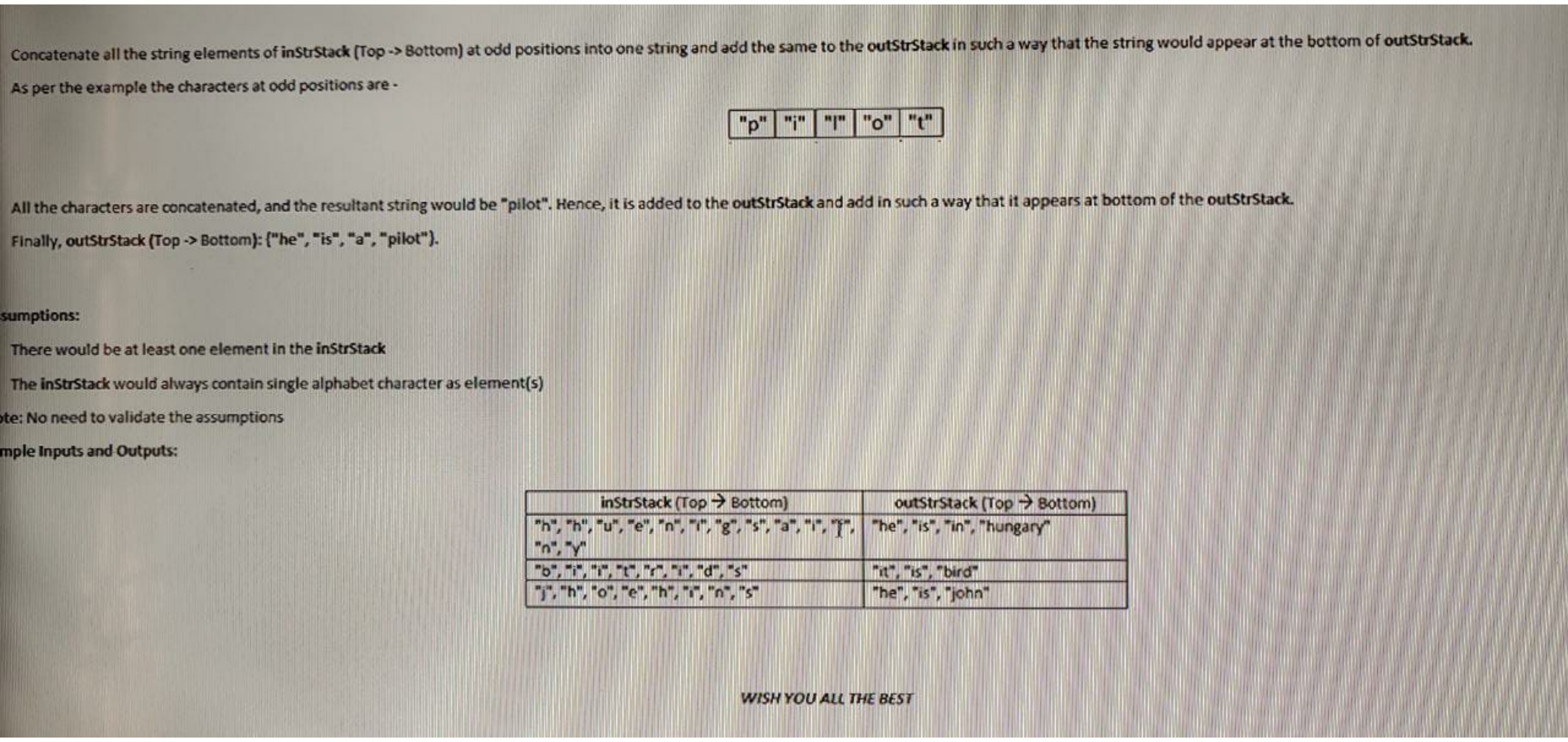
ssumptions:

There would be at least one element in the **inStrStack**

The **inStrStack** would always contain single alphabet character as element(s)

ote: No need to validate the assumptions

ample Inputs and Outputs:



Concatenate all the string elements of **inStrStack** (Top -> Bottom) at odd positions into one string and add the same to the **outStrStack** in such a way that the string would appear at the bottom of **outStrStack.**

As per the example the characters at odd positions are -

"p"	"i"	"l"	"o"	"t"

All the characters are concatenated, and the resultant string would be "pilot". Hence, it is added to the **outStrStack** and add in such a way that it appears at bottom of the **outStrStack.**

Finally, **outStrStack** (Top -> Bottom): ("he", "is", "a", "pilot").

sumptions:

There would be at least one element in the **inStrStack**

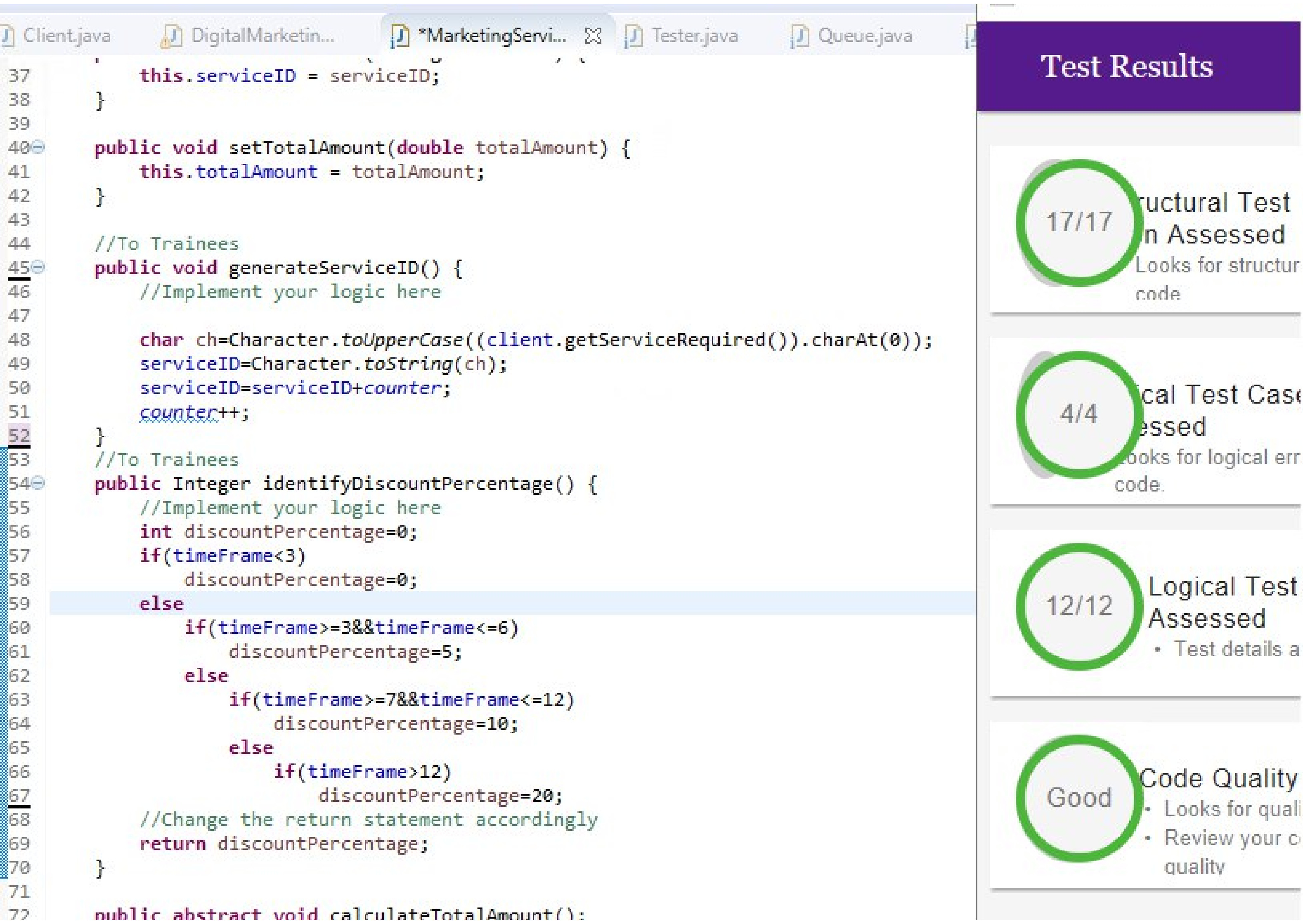
The **inStrStack** would always contain single alphabet character as element(s)

te: No need to validate the assumptions

mple Inputs and Outputs:

inStrStack (Top → Bottom)	outStrStack (Top → Bottom)
"h", "h", "u", "e", "n", "i", "g", "s", "a", "i", "r", "n", "y"	"he", "is", "in", "hungary"
"b", "i", "i", "t", "r", "i", "d", "s"	"it", "is", "bird"
"j", "h", "o", "e", "h", "i", "n", "s"	"he", "is", "john"

WISH YOU ALL THE BEST



```java
        this.serviceID = serviceID;
    }

    public void setTotalAmount(double totalAmount) {
        this.totalAmount = totalAmount;
    }

    //To Trainees
    public void generateServiceID() {
        //Implement your logic here

        char ch=Character.toUpperCase((client.getServiceRequired()).charAt(0));
        serviceID=Character.toString(ch);
        serviceID=serviceID+counter;
        counter++;
    }
    //To Trainees
    public Integer identifyDiscountPercentage() {
        //Implement your logic here
        int discountPercentage=0;
        if(timeFrame<3)
            discountPercentage=0;
        else
            if(timeFrame>=3&&timeFrame<=6)
                discountPercentage=5;
            else
                if(timeFrame>=7&&timeFrame<=12)
                    discountPercentage=10;
                else
                    if(timeFrame>12)
                        discountPercentage=20;
        //Change the return statement accordingly
        return discountPercentage;
    }

    public abstract void calculateTotalAmount();
```

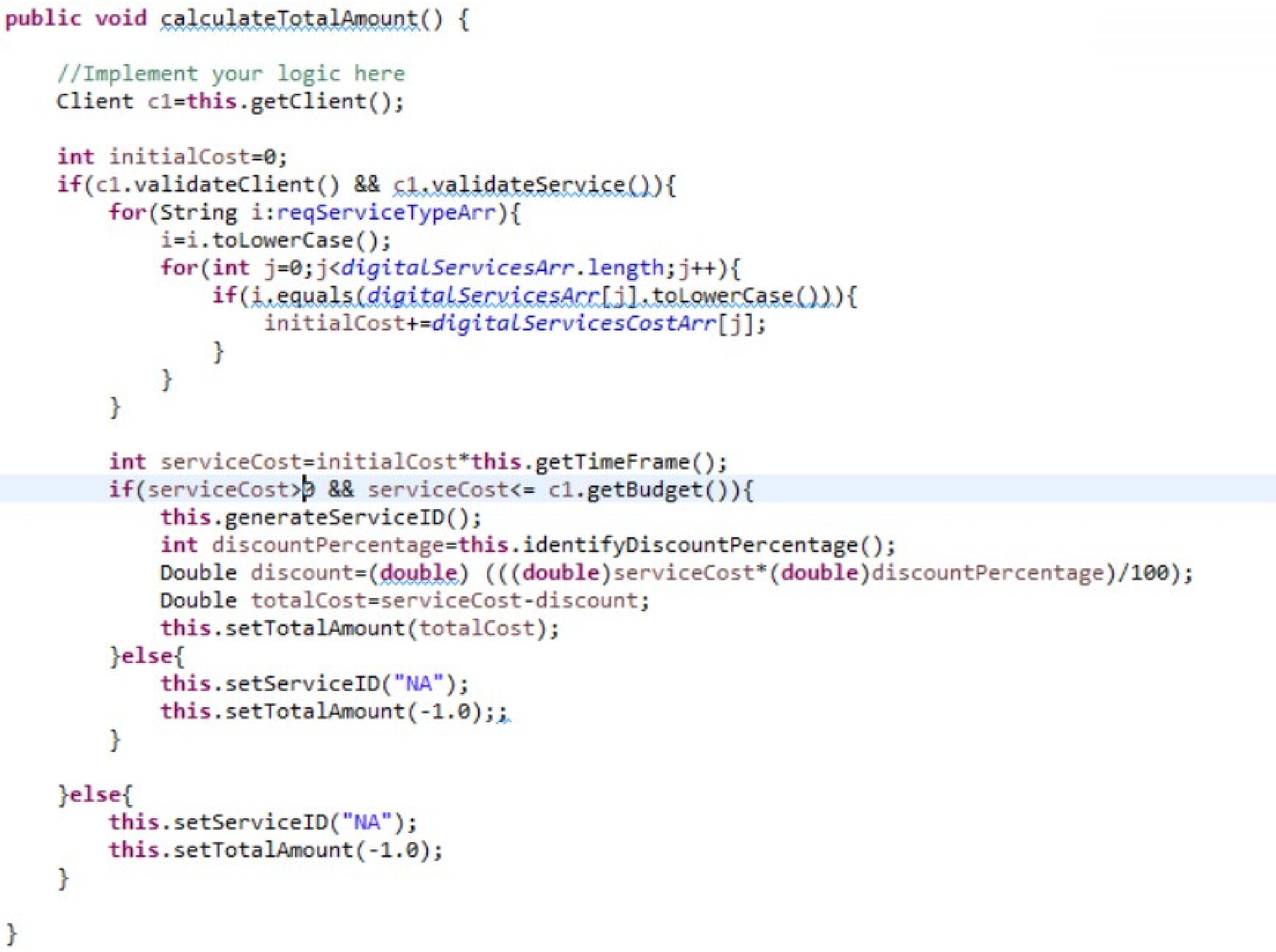
Test Results

17/17 ructural Test
n Assessed
Looks for structur
code

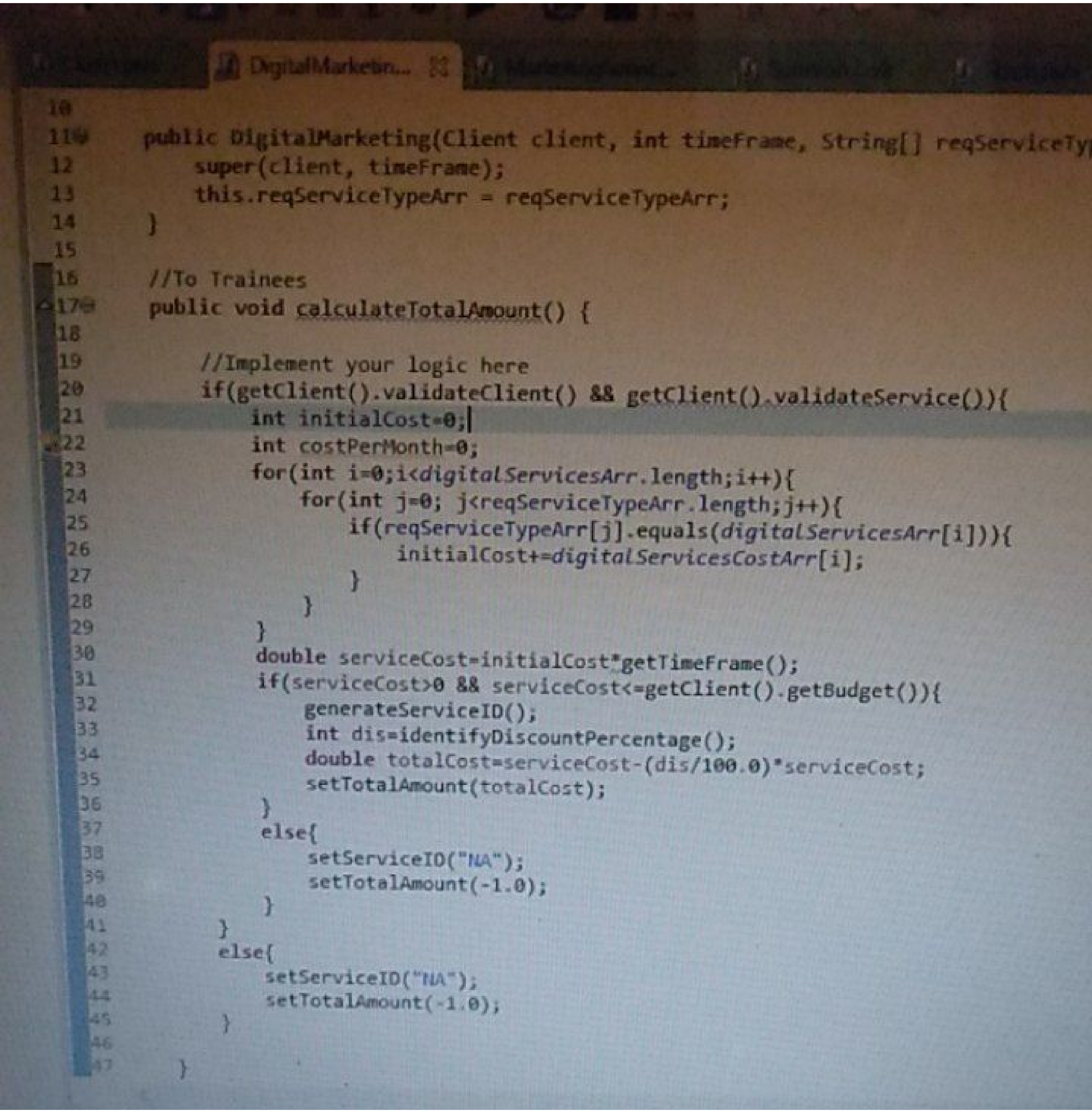
4/4 ical Test Case
essed
ooks for logical err
code.

12/12 Logical Test
Assessed
- Test details a

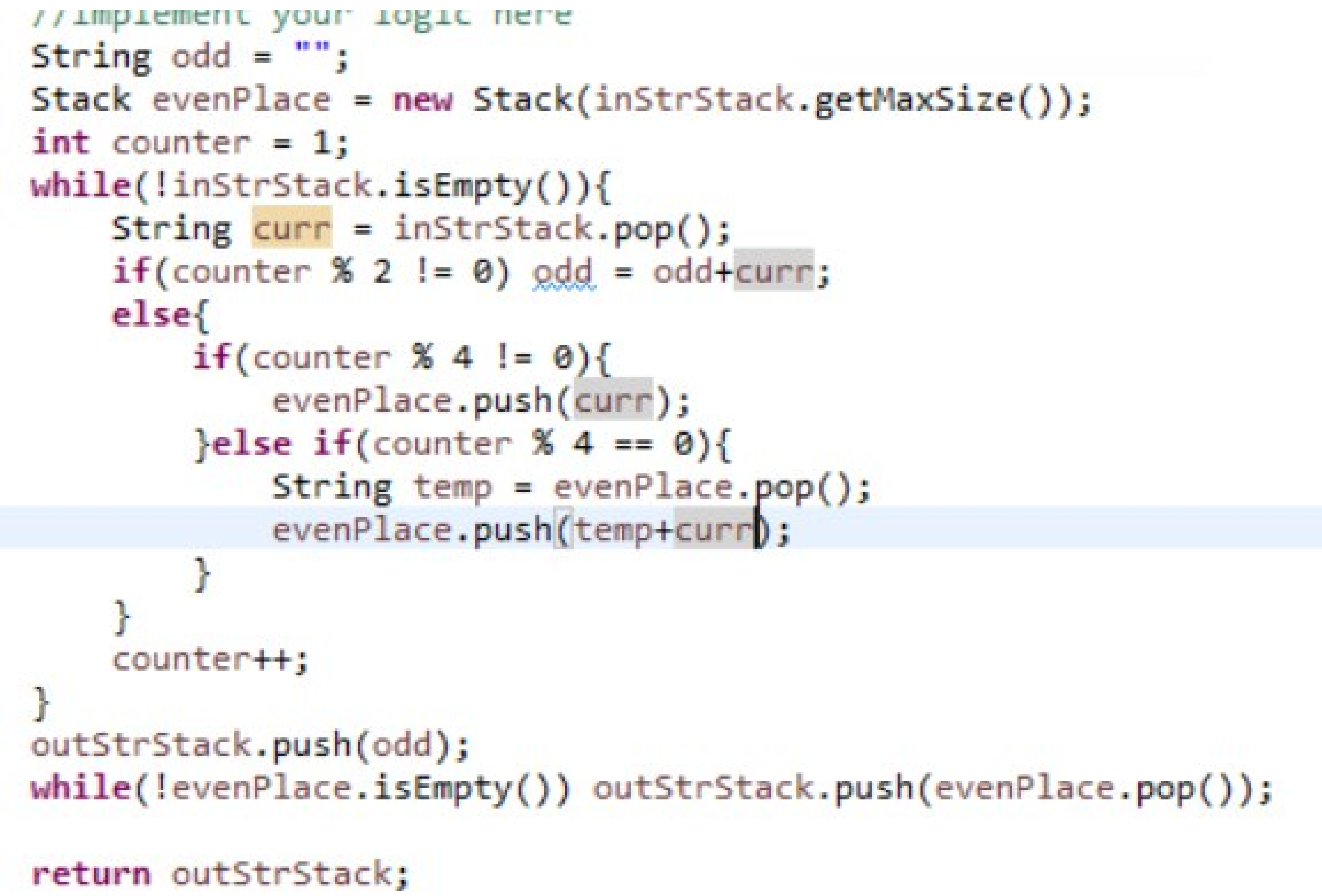
Good Code Quality
- Looks for quali
- Review your c
quality



```java
public void calculateTotalAmount() {

    //Implement your logic here
    Client c1=this.getClient();

    int initialCost=0;
    if(c1.validateClient() && c1.validateService()){
        for(String i:reqServiceTypeArr){
            i=i.toLowerCase();
            for(int j=0;j<digitalServicesArr.length;j++){
                if(i.equals(digitalServicesArr[j].toLowerCase())){
                    initialCost+=digitalServicesCostArr[j];
                }
            }
        }

        int serviceCost=initialCost*this.getTimeFrame();
        if(serviceCost>0 && serviceCost<= c1.getBudget()){
            this.generateServiceID();
            int discountPercentage=this.identifyDiscountPercentage();
            Double discount=(double) (((double)serviceCost*(double)discountPercentage)/100);
            Double totalCost=serviceCost-discount;
            this.setTotalAmount(totalCost);
        }else{
            this.setServiceID("NA");
            this.setTotalAmount(-1.0);;
        }

    }else{
        this.setServiceID("NA");
        this.setTotalAmount(-1.0);
    }

}
```



```java

public DigitalMarketing(Client client, int timeFrame, String[] reqServiceTy
    super(client, timeFrame);
    this.reqServiceTypeArr = reqServiceTypeArr;
}

//To Trainees
public void calculateTotalAmount() {

    //Implement your logic here
    if(getClient().validateClient() && getClient().validateService()){
        int initialCost=0;
        int costPerMonth=0;
        for(int i=0;i<digitalServicesArr.length;i++){
            for(int j=0; j<reqServiceTypeArr.length;j++){
                if(reqServiceTypeArr[j].equals(digitalServicesArr[i])){
                    initialCost+=digitalServicesCostArr[i];
                }
            }
        }
        double serviceCost=initialCost*getTimeFrame();
        if(serviceCost>0 && serviceCost<=getClient().getBudget()){
            generateServiceID();
            int dis=identifyDiscountPercentage();
            double totalCost=serviceCost-(dis/100.0)*serviceCost;
            setTotalAmount(totalCost);
        }
        else{
            setServiceID("NA");
            setTotalAmount(-1.0);
        }
    }
    else{
        setServiceID("NA");
        setTotalAmount(-1.0);
    }

}
```



```
//implement your logic here
String odd = "";
Stack evenPlace = new Stack(inStrStack.getMaxSize());
int counter = 1;
while(!inStrStack.isEmpty()){
    String curr = inStrStack.pop();
    if(counter % 2 != 0) odd = odd+curr;
    else{
        if(counter % 4 != 0){
            evenPlace.push(curr);
        }else if(counter % 4 == 0){
            String temp = evenPlace.pop();
            evenPlace.push(temp+curr);
        }
    }
    counter++;
}
outStrStack.push(odd);
while(!evenPlace.isEmpty()) outStrStack.push(evenPlace.pop());

return outStrStack;
```