DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE

Course: CS210 - DATA STRUCTURES AND ALGORITHMS

Instructor: Dr. Alaa Shamasneh

CS210 Project

Section 200 & 204

Prepared by:

Hifaa Zin Eddin

220410581

Maryah Alrebdi

220511210

Phase 1

Part 1 (Stack & Queue):

1-

```
package cs210Project;

public class Queue <G> {
//*Nested Node class SinglyLinkedList Data Structure implementation
	class Node<G>{

		private G data;
		private Node<G> next;


		public Node() {
			data = null;
			next = null;

		}

		public Node (G d, Node<G> n) {
			data = d;
			next = n;

		}}//*endNodeClass

		//*Queue LinkedList Data Structure implementation
		private Node<G> front,rear;
		private int size;
```

```
public Queue() {
	front = rear = null;
	size = 0;
}

//*Method isEmpty()
public boolean isEmpty() {
	return size == 0;
}

//*Method size()
public int size() {
	return size;
}

//*enqueue()
public void enqueue(G e) {
	if (rear == null) {
		front = rear = new Node<G>(e,null);

	}else {
		rear.next = new Node<G>(e,null);
		rear = rear.next;

	}
	size++;
}

//*deQueue()
public G deQueue() {
```

```
        if(front==null) {
                return null;}else {
        G temp = front.data;
        front = front.next;
        size--;
        if(front == null)
                rear = null;
        return temp;    }
}

//*Print() for traversing
public void Print() {
        if(size == 0)
                System.out.println(" *The Queue is Empty!* ");
        else {
        Node <G> pointer = front;
        while(pointer != null) {//O(n)
                System.out.println(pointer.data);
                pointer = pointer.next;
        }}

}
//Main Test
public static void main (String [] args) {
        Queue q = new Queue();
        q.enqueue(10);
        q.deQueue();
        q.enqueue(11);
        q.deQueue();
        q.enqueue(12);
        q.deQueue();
```

```
                System.out.println("The contents of Queue q: ");
                q.Print();

                Queue q1 = new Queue();
                System.out.println("The contents of Queue q1: ");
                q1.Print();

                Queue q2 = new Queue();
                q2.enqueue("CS210");
                q2.enqueue("CS285");
                System.out.println("Items before dequeue: ");
                q2.Print();
                if(!q2.isEmpty()) {
                q2.deQueue();
                System.out.println("Items after dequeue: ");
                q2.Print();}else {
                        System.out.println("The queue is empty, nothing to dequeue.");
                }


        }//*endMainMethod
}//*endQueueClass
```

```
    public static void main(String[] args) {
       Queue q = new Queue();
                        q.enqueue(10);
                        q.deQueue();
                        q.enqueue(11);
                        q.deQueue();
                        q.enqueue(12);
                        q.deQueue();

                        System.out.println("The contents of Queue q: ");
                        q.Print();

                        Queue q1 = new Queue();
                        System.out.println("The contents of Queue q1: ");
                        q1.Print();

                        Queue q2 = new Queue();
                        q2.enqueue("CS210");
                        q2.enqueue("CS285");
                        System.out.println("Items before dequeue: ");
                        q2.Print();
                        if(!q2.isEmpty()) {
                        q2.deQueue();
                        System.out.println("Items after dequeue: ");
                        q2.Print();}else {
                                System.out.println("The queue is empty, nothing to dequeue.");
                        }
```

put - cs210project (run)

```
run:
The contents of Queue q:
 *The Queue is Empty!*
The contents of Queue q1:
 *The Queue is Empty!*
Items before dequeue:
CS210
CS285
Items after dequeue:
CS285
BUILD SUCCESSFUL (total time: 0 seconds)
```

2-

```
package cs210Project;

public class Stack<G> {
//*Nested Node Class SinglyLinkedList Data Structure implementation
        class Node<G>{

                private G data;
                private Node<G> next;

                public Node() {
                        data = null;
```

```
        next = null;
}

public Node (G d, Node<G> n) {
        data = d;
        next = n;

}}//*endNodeClass

//*Stack Data Structure
private Node<G> top;
private int size;

public Stack() {
        top = null;
        size = 0;

        }

//*Method size()
public int size() {
        return size;

        }

//*Method isEmpty
public boolean isEmpty() {
        return size == 0;
        }

//*Push()at top FILO like addfirst() in LL
```

```
public void push(G e) {
        Node<G> newNod= new Node<G>(e,null);
        newNod.next=top;
        top=newNod;
        size++;
        }


//*Pop()at last like removefirst() in LL
public G pop() {
if(top == null && size==0) {
return null;
        }else {
                G e = top.data;
                top = top.next;
                size--;
                return e;
                }
        }


//*Peek() returns first element at top without removing it
public G peek() {
        if(top == null) {
                return null;
        }else {
                G e = top.data;
                return e;
                }
        }
```

```
//*Method Print to traverse
public void Print() {
        if(size == 0) {
                System.out.println(" *The Stack is Empty!* ");
        }else {
                Node <G> pointer = top;
                while(pointer != null) { //O(n)
                        System.out.println(pointer.data);
                        pointer = pointer.next;
                        }
                        }
        }

//*Main Test
public static void main(String[]args) {
        Stack s = new Stack();
        s.push(10);
        s.pop();
        s.push(11);
        s.pop();
        s.push(12);
        s.pop();
        s.push(13);
        s.pop();
        s.push(14);
        s.pop();
        System.out.println("Return element at the top: ");
        if(s.peek()==null)
                System.out.println("There is no element at the top of stack!!");
        else System.out.println("The element at the top of stack s is: ");
```

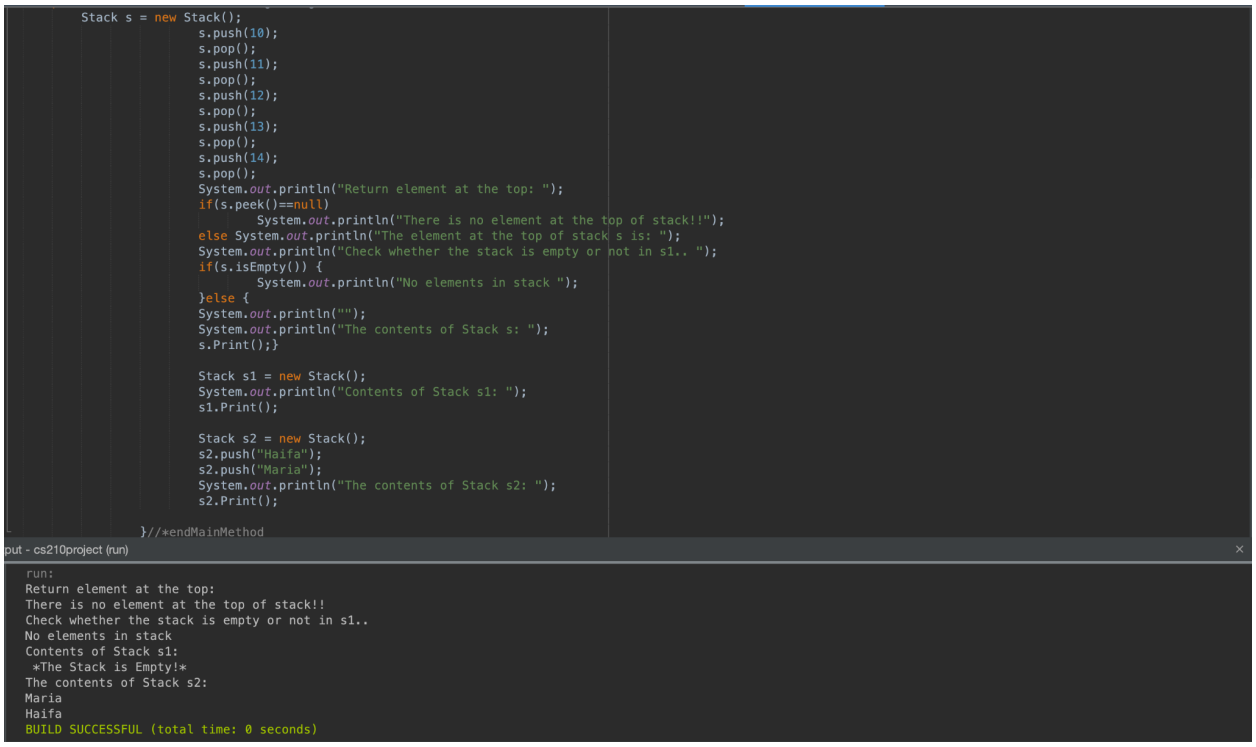
```
                System.out.println("Check whether the stack is empty or not in s1.. ");
                if(s.isEmpty()) {
                        System.out.println("No elements in stack ");
                }else {
                System.out.println("");
                System.out.println("The contents of Stack s: ");
                s.Print();}

                Stack s1 = new Stack();
                System.out.println("Contents of Stack s1: ");
                s1.Print();

                Stack s2 = new Stack();
                s2.push("Haifa");
                s2.push("Maria");
                System.out.println("The contents of Stack s2: ");
                s2.Print();

        }//*endMainMethod
}//*endStackClass
```

Time Complexity for Both Classes (Stack & Queue) is: O(N).



```
        Stack s = new Stack();
                        s.push(10);
                        s.pop();
                        s.push(11);
                        s.pop();
                        s.push(12);
                        s.pop();
                        s.push(13);
                        s.pop();
                        s.push(14);
                        s.pop();
                        System.out.println("Return element at the top: ");
                        if(s.peek()==null)
                                System.out.println("There is no element at the top of stack!!");
                        else System.out.println("The element at the top of stack s is: ");
                        System.out.println("Check whether the stack is empty or not in s1.. ");
                        if(s.isEmpty()) {
                                System.out.println("No elements in stack ");
                        }else {
                        System.out.println("");
                        System.out.println("The contents of Stack s: ");
                        s.Print();}

                        Stack s1 = new Stack();
                        System.out.println("Contents of Stack s1: ");
                        s1.Print();

                        Stack s2 = new Stack();
                        s2.push("Haifa");
                        s2.push("Maria");
                        System.out.println("The contents of Stack s2: ");
                        s2.Print();

                }//*endMainMethod
```

```
put - cs210project (run)
  run:
  Return element at the top:
  There is no element at the top of stack!!
  Check whether the stack is empty or not in s1..
  No elements in stack
  Contents of Stack s1:
   *The Stack is Empty!*
  The contents of Stack s2:
  Maria
  Haifa
  BUILD SUCCESSFUL (total time: 0 seconds)
```

Part 2 (Tower of Hanoi):

```
package cs210Project;


public class Stack_ {
//Data structure of the stack
                int size;
                int top;
                int Disks[];

                //To implement the size of the stack to the gives rodes

                Stack_ StackData(int size) {
                Stack_ s = new Stack_();
```

```
s.size = size;
s.top=-1;
s.Disks=new int[size];
return s;
}

//check if the stack is full based the top determined by the size
public boolean isFull(Stack_ s) {
	return s.top == s.size-1;
}

//check is stack is empty from the top of the stack only
public boolean isEmpty(Stack_ s) {
	return s.top==-1;
}

//Push() method to insert disks into the stack rodes
public void push(Stack_ s, int value) {
	if(isFull(s))
		return;
	s.top++;
	s.Disks[s.top]=value;
}

//Pop() method to return and remove the top element from the stack
public int pop(Stack_ s) {
	if(isEmpty(s))
		return Integer.MIN_VALUE;//constant integer that returns the least value
	s.top--;
	return s.Disks[s.top+1];
}
```

```
public int Top(Stack_ s) {
	return s.Disks[s.top];
}

//Methods to move disks between the 3 rodes
void RodeMovements(Stack_ source, Stack_ destination, char sourc, char destinatio) {
	//initialize two rodes in order to move between them
	int Rode1Top = pop(source);
	int Rode2Top = pop(destination);

	//if the first rode is empty move top disks
	if(Rode1Top==Integer.MIN_VALUE) {
		push(source,Rode2Top);
		DisplayMovements(destinatio, sourc, Rode2Top);
	}

	//if the second rode is empty move top disk
	else if(Rode2Top==Integer.MIN_VALUE) {
		push(destination, Rode1Top);
		DisplayMovements(sourc, destinatio, Rode1Top);
	}

	//if the top of the first rode is GREATER than the top of the second rode then move disks
	else if(Rode1Top>Rode2Top) {
		push(source, Rode1Top);
		push(source, Rode2Top);
		DisplayMovements(destinatio, sourc, Rode2Top);

	}
```

```
        //if the top disk of the first rode is LESS than the top of the second rode then move disks
        else {
                push(destination, Rode2Top);
                push(destination, Rode1Top);
                DisplayMovements(sourc, destinatio, Rode1Top);
        }
}

//Method to print the disk movements of the 3 rodes
void DisplayMovements(char s, char d, int data) {
        System.out.println("Move the disk data " + data + " from " + s +" to " + d);
}

//Method to print the content of the stack at the beginning
void DisplayContent(Stack_ source, Stack_ additionalRode, Stack_ destination) {
        int r;
        System.out.print("\nThe source rode: ");
        for(r=source.top;r>=0;r--)
                System.out.print(source.Disks[r] + " ");
        System.out.println();

        System.out.print("\nThe additional rode: ");
        for(r=additionalRode.top;r>=0;r--)
                System.out.print(destination.Disks[r]+" ");
        System.out.print("\nThe destination rode: ");
        for(r=destination.top;r>=0;r--)
                System.out.print(destination.Disks[r]+" ");
        System.out.println();
}

void DiskIterations(int diskNum, Stack_ source, Stack_ additionalRode, Stack_ destination) {
```

```
int r;
int NumOfMoves;
char sourc = 'A';
char destinatio = 'B';
char additional = 'C';

//Method to interchange between the destination rode and additional rode if the number of disks are even
if(diskNum%2 == 0) {
	char temp = destinatio;
	destinatio = additional;
	additional = temp;
}

//Calculation of the total number of moves between each rode which is 2^n-1
NumOfMoves=(int)(Math.pow(2, diskNum)-1);

for(r=diskNum;r>=1;r--)
	push(source,r);

System.out.println("The Tower of Hanoi before movements: ");
DisplayContent(source, additionalRode, destination);
System.out.println();

for(r=1;r<=NumOfMoves;r++) {//for the total number of moves

	//if r to the 3 rodes is in the rode 1 then it moves between char A and B then it moves to rode 2 and repeats if r is 2 until total
	//and repeats if the total number of moves is 2 then it moves between char B and C until total number of moves is 7
```

```
                    if(r%3 == 1)
                        RodeMovements(source, destination, sourc, destinatio);
                    else if(r%3 == 2)
                        RodeMovements(source, additionalRode, sourc, destinatio);
                    else if(r%3==0)
                        RodeMovements(additionalRode, destination,additional,destinatio);

                }
                System.out.println("\nThe Tower of Hanoi after movements: ");
                DisplayContent(source, additionalRode, destination);


        }
}

public class Tooo {

        public static void main(String[]args) {
            //Tower of Hanoi main method using stack
            int diskNum=3;
            Stack_ d = new Stack_();//the object representing the stack
            Stack_ source;
            Stack_ destination;
            Stack_ additionalRode;

            //initialize three stacks of the number of disks as its size
            source = d.StackData(diskNum);
            destination = d.StackData(diskNum);
            additionalRode = d.StackData(diskNum);
            d.DiskIterations(diskNum, source, additionalRode, destination);
        }
```

}

Time complexity: O(2^N)

```
run:
The Tower of Hanoi before movements:

The source rode: 1 2 3

The additional rode:
The destination rode:

Move the disk data 1 from A to B
Move the disk data 2 from A to C
Move the disk data 1 from B to C
Move the disk data 3 from A to B
Move the disk data 1 from C to A
Move the disk data 2 from C to B
Move the disk data 1 from A to B

The Tower of Hanoi after movements:

The source rode:

The additional rode:
The destination rode: 1 2 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

Phase 2

(Real Life Situation - Customers at a Grocery Store Check Out):

```
package cs210Project;
import java.util.*;
public class GroceryStoreApplication {
                //Cart class to create objects
public static class Cart{
```

```
		private int CartNO;//attribute num of carts
		private Stack<String> itemName = new Stack();//attribute stack to enter items; name
		private Stack<Double> itemPrice = new Stack();//attribute stack to enter items' price
		public Cart() {
			CartNO = 0;
		}
}//end Cart class
static Scanner userInput = new Scanner(System.in); //in case I needed to use outside main method
public static void main (String [] args) {
		Queue<Cart> customers = new Queue<>();//queue of type Cart obj
		double GrandTotal = 0;
		Queue<Double> donations = new Queue<>();//queue of donations type double
		double DonationsTotal = 0;

		int choice;
		do {
			DisplayFeatures();//call method to display features of program
			choice = userInput.nextInt();
			System.out.println("----------------------");
				switch (choice) {
				case 1:
					System.out.print("Enter Number of Items in Customer's Cart: ");
					int itemsNum = userInput.nextInt();
					Cart cart = new Cart();//create new obj
					cart.CartNO++;//Increment number of carts in cart object when new customer
		checks out
					for(int k = 1; k <= itemsNum; k++) {//for loop to enter the items' info based on
		the number of items the user selected
						System.out.println("Enter Item [ " + k + " ]'s Info: ");
						System.out.print("Item's Name: ");
						String itemName = userInput.next();
```

```
                    System.out.print("Item's Price: ");
                    double itemPrice = userInput.nextDouble();
                    cart.itemName.push(itemName);
                    cart.itemPrice.push(itemPrice);
                }
                customers.enqueue(cart);//enqueue obj in queue to keep record
                System.out.println("----------------------");
                System.out.println("Here is Your Bill: ");
                System.out.println("----------------------");
                double total = 0;
                System.out.println("Kindly Pay For The Following Items: ");
                System.out.println("");
                System.out.println("----------------------");
                while(! cart.itemName.isEmpty()  &&  ! cart.itemPrice.isEmpty()) {//traversing
while validating a not empty stack
                    total = total + cart.itemPrice.peek();//gettinng each items price from
stack without deleting it and adding it to get total
                    System.out.println(cart.itemName.pop() + " ->->->->->->->->->->->->-> "
+ cart.itemPrice.pop());
                    System.out.println("");
                }//end while loop
                System.out.println("The Total is: " + total + "SR");
                System.out.println("Thank You For Shopping With Us ! ");
                GrandTotal = GrandTotal + total;//Total of all earnings for every customer that
checked out
                System.out.println("");
                System.out.println("Next Customer Please. ");
                break;
            case 2:
                System.out.println("Total Number of Earnings is: " + GrandTotal + "SR");
                break;
```

```
                case 3:
                        System.out.println(customers.size() + " Customers Have Visited Muheet
Altowfeer. ");//number of people who shopped
                        break;
                case 4:
                        System.out.print("Enter The Amount of Money You Would Like To Donate: ");
                        double don = userInput.nextDouble();
                        System.out.println("Thank You For Donating " + don + "SR");
                        donations.enqueue(don);//enqueue donations
                        break;
                case 5:
                        System.out.println(donations.size() + " Donations Have Been Made!");//returns
number or people who donated based on queue's size
                        break;
                case 6:
                        while(!donations.isEmpty()) {
                                DonationsTotal = DonationsTotal + donations.deQueue();    //total
donations money summed
                        }//end while loop
                        System.out.println("The Total of Donations Money is: " + DonationsTotal +
"SR");
                        break;
                case 7:
                        System.out.println("You Have Exited the System.");
                        System.exit(0);
                        break;
                        default:
                                System.out.printf("Invalid Choice, Try Again!\n");
                }
        }while(choice != 7);//end DoWhile Loop O(n)
}//end MainMethod
```

```
public static void DisplayFeatures() {
		System.out.println("----------------------");
		System.out.println("Wlecome To Muheet Altowfeer System!");
		System.out.println("1.) Check Out.");
		System.out.println("2.) Find Out Total Number of Earnings.");
		System.out.println("3.) Find Out Number of Customers Who Visited.");
		System.out.println("4.) Donate to Charities.");
		System.out.println("5.) Find Out Number of Donations Made.");
		System.out.println("6.) Find Out Total Amount of Money that Was Donated.");
		System.out.println("7.) Exit System.");
		System.out.print("Enter Your Request: ");
		return;
		}//endDisplayFeaturesMethod
}//endGroceryStoreApplication
```

Time Complexity : O(N^2)

Note: Stack and Queue classes in part 1 of phase one were used to initialize objects in the real life application.

```
---------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 8
---------------------
Invalid Choice, Try Again!
---------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request:
```

```
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 1
--------------------
Enter Number of Items in Customer's Cart: 2
Enter Item [ 1 ]'s Info:
Item's Name: Tea
Item's Price: 30
Enter Item [ 2 ]'s Info:
Item's Name: Coffee
Item's Price: 2.95
--------------------
Here is Your Bill:
--------------------
Kindly Pay For The Following Items:

--------------------
Coffee ->->->->->->->->->->->->-> 2.95

Tea ->->->->->->->->->->->->-> 30.0

The Total is: 32.95SR
Thank You For Shopping With Us !

Next Customer Please.
--------------------
```

```
Next Customer Please.
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 1
--------------------
Enter Number of Items in Customer's Cart: 3
Enter Item [ 1 ]'s Info:
Item's Name: Sugar
Item's Price: 4
Enter Item [ 2 ]'s Info:
Item's Name: Rice
Item's Price: 50
Enter Item [ 3 ]'s Info:
Item's Name: Milk
Item's Price: 20.95
--------------------
Here is Your Bill:
--------------------
Kindly Pay For The Following Items:

--------------------
Milk ->->->->->->->->->->->->-> 20.95

Rice ->->->->->->->->->->->->-> 50.0

Sugar ->->->->->->->->->->->->-> 4.0

The Total is: 74.95SR
Thank You For Shopping With Us !

Next Customer Please.
```

```
Next Customer Please.
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 2
--------------------
Total Number of Earnings is: 107.9SR
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 3
--------------------
2 Customers Have Visited Muheet Altowfeer.
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 4
--------------------
Enter The Amount of Money You Would Like To Donate: 400
Thank You For Donating 400.0SR
--------------------
```

```
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 4
--------------------
Enter The Amount of Money You Would Like To Donate: 30
Thank You For Donating 30.0SR
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 4
--------------------
Enter The Amount of Money You Would Like To Donate: 20
Thank You For Donating 20.0SR
--------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 5
--------------------
3 Donations Have Been Made!
--------------------
```

```
Enter The Amount of Money You Would Like To Donate: 20
Thank You For Donating 20.0SR
---------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 5
---------------------
3 Donations Have Been Made!
---------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 6
---------------------
The Total of Donations Money is: 450.0SR
---------------------
Wlecome To Muheet Altowfeer System!
1.) Check Out.
2.) Find Out Total Number of Earnings.
3.) Find Out Number of Customers Who Visited.
4.) Donate to Charities.
5.) Find Out Number of Donations Made.
6.) Find Out Total Amount of Money that Was Donated.
7.) Exit System.
Enter Your Request: 7
---------------------
You Have Exited the System.
```