Assignment1

=Client change=

Author: Ricardo Laner Jacobsen Teixeira (34031229)

Date: 08/09/2020

Filename: Assignment

**PUROSE:** This program was created with the intuition of helping the user on giving change based on a specific amount.

**2. REQUIREMENTS/SPECIFICATION:** The program should first ask how many people the user wishes to deal with along it. Once the quantity is given, then it is requested to start inputting the data about each customer. At the end of each client, a message is displayed asking the user if he/she wants to continue entering data.

When the total quantity of clients gets to the end, the program will prompt a menu offering some options to the user.

There are five options in which number five terminates the program.

**3.USER GUIDE:** In Netbeans goto file/ Import project / From zip / Select the directory where you downloaded the file. Now it is imported. Press the shortcut f6 or click on  to run the program.

**4.STRUCTURE/DESIGN/ALGORITHM:**

**<Change class>**

A. Declares the variable personNmae, coinAmount, int fiftyCent, tenCent, fiveCent, totalFifty, totalTen, totalFive and also declares a object array.

B. Change() - Defaualt constructor

C. Change(String name, int amount) - Constructor setting the values to variables

D. Change(int size) - Constructor receiving the size parameter for creating the array with the passed size

E. findPerson(String name) - Check if the name is already into the array and return if the name was found or not in form of position(case -1 not found)

F. addChange(String name, int amount)- Set the name and the amount of the object

G. setPersonName(String name) - sets the client object name.

H. calculateChange(int amount) - based on the amount of the client found calculates the change denominations.

I. smallestAmount() – Loops from the beginning to the end of the array to find the smallest amount.

J. largestAmount() - Loops from the beginning to the end of the array to find the larges amount.

K. showTotal() – Calculates the total of each denomination from all the customers of the array.

L. setCoinValue(int coin) - sets the client object amount

M. addCoinValue(int coin)- If the user wants to cadaster a customer that already is in the array this method will add only the new amount + the old amount.

N. getPersonName() – Returns the customer name .

O. getCoinValue() – Returns the customer value.

P. keepEntering (char resp) – Receive the response of the user and return if continues inputting data or not.

Q. hardCode() – In case the teacher decides to test the program to make it easy this method will auto cadaster 12 customer into the program.

**<Client class>**

A. declares variables c, name,amount, keep, autoInput, inRange, howMany, int menu.

B. Display the student information.

C. Display the initial statements.

D. Get the number of people to be entered.

E. Loops till the end of the number of people entered inputting the data.

F. Display the menu

G. Ask the user option

H. Operates according to the user choice

I. Quit when the user enters option number 5

**PSEUDUCODE**

**<CLIENT CLASS>**

*BEGIN*

public static Scanner keyboard = new Scanner (System.in);

static int c=0;

String name=""

int amount=0;

boolean keep=true;

char autoInput=' ';

boolean inRange=false;

int howMany =0;

int menu = 0;

-StudentInfo()

DISPLAY: “Name:Ricardo Laner JacobsenTeixeira”

DISPLAY: “Student Number: 34031229”

DISPLAY: “Mode of enrolment: Internal”

DISPLAY: “Tutor: Upeke”

DISPLAY: “Tutorial: Thursday 3:30 pm”

-displayInitialStatements();

DISPLAY: FOR TESTING the program AND automatically complete the 12 clients for you?

DISPLAY: WHERE IT ASKS THE NUMBER OF PERSON ENTER THE NUMBER -1

DISPLAY: RECOMENDED: Please enter at least 12 records to test the program.

DISPLAY: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DYSPLAY: Please enter number of people:

howMany= keyboard.nextInt()

inRange = false

Change person = new Change(howMany);

WHILE(keep==true )

IF(howMany==-1)

c=12;

break

End of IF

DISPLAY: Please enter the PERSON NAME:

name=keyboard.nextLine()

String provisoryName=" “

LOOP FOR FROM 0 TO name.length i++)

IF(name.charAt(i)==' ')

name = provisoryName

break

ELSE

provisoryName+=name.charAt(i)

WHILE(inRange==false)

DISPLAY: Please enter the COIN VALUE for the person (RANGE 5 TO 95, MULTIPLE of 5):

amount=keyboard.nextInt()

IF(amount>=5 && amount<=95 && amount%5==0)

inRange=true

ELSE

inRange=false

end of IF

end of WHILE

int ret = person.findPerson(name)

IF(c==0){person.addChange( name, amount)

ELSE IF(ret!=-1)

person.peep[ret].addCoinValue(amount)

c--

ELSE IF(ret==-1)

person.addChange( name, amount)

c++

IF(c==howMany)

break

DISPLAY: Do you wish to KEEP ENTERING? [y/n]: ");

keep = person.keepEntering(keyboard.next().toUpperCase().charAt(0))

keyboard.nextLine()

inRange = false

-displayMenu()

. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Change Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DISPLAY: 1. Enter a name and display change to be given for each denomination.

DISPLAY: 2. Find the name with the smallest amount and display change to be given for each denomination.

DISPLAY: 3. Find the name with the largest amount and display change to be given for each denomination.

DISPLAY: 4. Calculate and display the total number of coins for each denomination, and the sum of these totals.

` DISPLAY: 5. Exit.

DISPLAY: What is your choice:

menu = keyboard.nextInt()

WHILE(menu>5 || menu<1)

DISPLAY: The option you entered is INVALID TRY AGAIN

End WHILE

-displayMenu()

menu = keyboard.nextInt()

WHILE( menu>=1 )

SWITCH(menu)

Case 1:

DISPLAY: Please enter the name of the person:

name = keyboard.next()

int personFound=person.findPerson(name)

IF(personFound==-1)

Name: name Not found

ELSE

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue())

DISPLAY: Customer + person name + amount +cents

DISPLAY: Change:

DISPLAY: 50 cents:

DISPLAY: 10 cents:

DISPLAY: 5 cents:

End of IF ELSE statement

Case 2:

personFound=person.smallestAmount()

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue()

DISPLAY: The name with the SMALLEST amount is person

DISPLAY: Change:

DISPLAY: 50 cents:

DISPLAY: 10 cents:

DISPLAY: 5 cents:

Case 3:

personFound=person.largestAmount()

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue()

DISPLAY: The name with the LARGEST amount is person

DISPLAY: Change:

DISPLAY: 50 cents:

DISPLAY: 10 cents:

DISPLAY: 5 centS:

case 4:

person.showTotal()

DISPLAY: The TOTAL OF FIFTY CENTS IS: " + person.totalFifty

DISPLAY: The TOTAL OF TEN CENTS IS: " +person.totalTen

DISPLAY: The TOTAL OF FIVE CENTS IS: " +person.totalFivE

DISPLAY: The sum of all clients is: " + (person.totalFifty+person.totalTen+person.totalFive)

person.totalFifty=0;

person.totalTen=0;

person.totalFive=0;

case 5:

DISPLAY: QUIT, thank you for using this program.\nBye!

System EXIT

End of SWTICH

displayMenu();

menu = keyboard.nextInt()

WHILE(menu>5 || menu<1)

DISPLAY: The option you entered is INVALID TRY AGAIN.

displayMenu()

menu = keyboard.nextInt()

End WHILE

*FINISH*

**<CHANGE CLASS>**

*BEGIN*

private String personName

private int coinAmount

int fiftyCent

int tenCent

int fiveCent

int totalFifty

int totalTen

int totalFive

Change peep[]

-Change()

SET THE CONSTRUCTOR AS DEFAULT

personName="Unknown"

coinAmount=0

- Change(String name, int amount)

SET THE CONSTRUCTOR WITH TWO PARAMETERS

this.personName = name

this.coinAmount = amount

-Change(int size)

IF size==-1

size=12

peep =new Change[size]

this.hardCode()

ELSE

peep =new Change[size]

- findPerson(String name)

int position=-1

FOR LOOP from 0 to Client.c i++)

IF(peep[i].personName.equalsIgnoreCase(name))

position= i

return position

-addChange(String name, int amount)

peep[Client.c] = new Change(name, amount)

-setPersonName(String name)

this.personName=name

-setPersonName(String name)

this.personName=name

-smallestAmount()

int smallest = peep[0].coinAmount

int position = 0;

FOR LOOP FORM 1 TO Client.c c++

IF(peep[c].coinAmount < smallest)

smallest = peep[c].coinAmount

position = c

return position

-largestAmount()

int largest = peep[0].coinAmount

` int position = 0

FOR LOOP FROM 1 TI Client.c c++

IF (peep[c].coinAmount > largest)

largest = peep[c].coinAmount;

position = c

-showTotal()

FOR LOOP FROM 0 TO Client.c i++

peep[i].calculateChange(peep[i].getCoinValue())

totalFifty+=peep[i].fiftyCent

totalTen+=peep[i].tenCent

totalFive+=peep[i].fiveCent

-setCoinValue(int coin)

this.coinAmount=coin

-addCoinValue(int coin){

this.coinAmount+=coin

- getPersonName()

return this.personName

-getCoinValue()

return this.coinAmount

-keepEntering (char resp)

boolean result=false

IF (resp=='Y')

result = true;

ELSE if (resp=='N')

result = false

return result

-hardCode()

peep[0] = new Change(Jane,20)

peep[1] = new Change(John,15)

peep[2] = new Change(Fred, 95)

peep[3] = new Change(Wilma, 85)

peep[4] = new Change(Paul, 65)

peep[5] = new Change(Marry,50)

peep[6] = new Change(Adam,75)

peep[7] = new Change(Xavier,70)

peep[8] = new Change(Ricardo, 30)

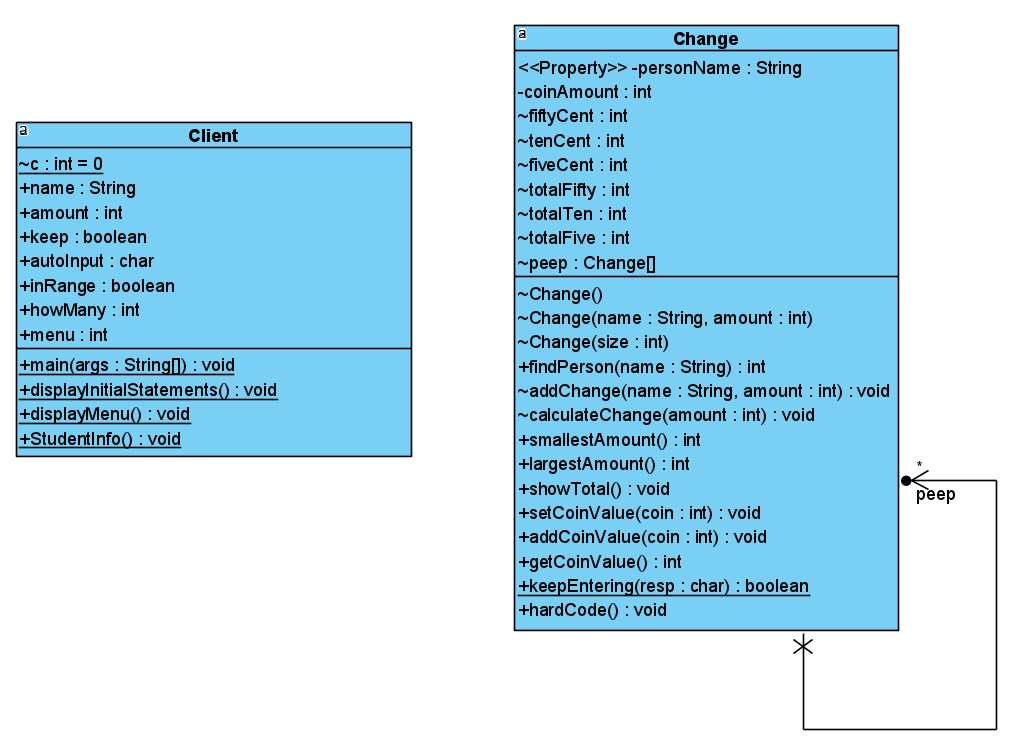
peep[9] = new Change(Camila, 65)

peep[10] = new Change(Martin,55);

peep[11] = new Change(Tom, 80)

*FINISH*

**<UML DIGRAM CLASS>**

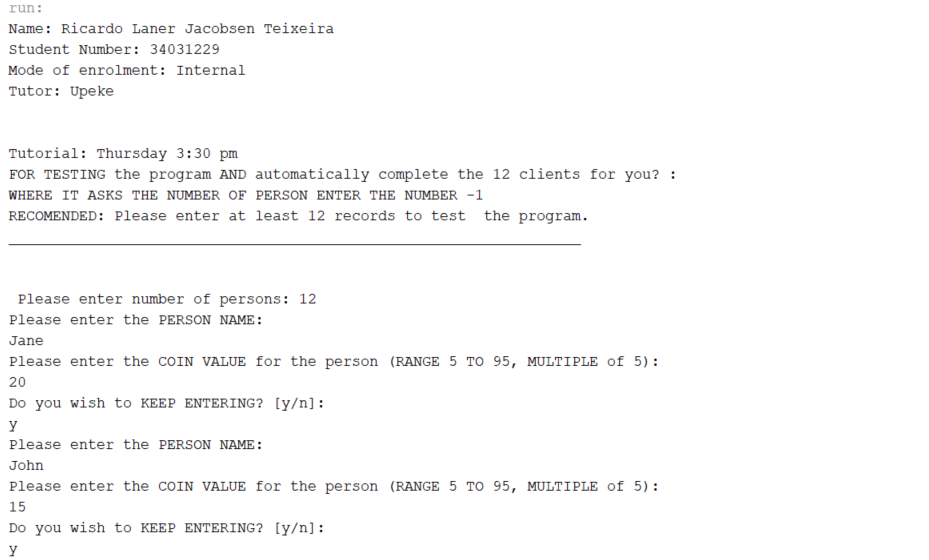


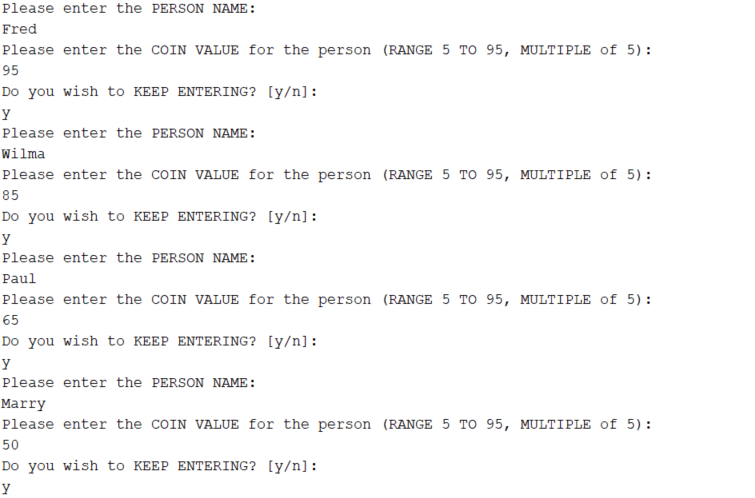
**5.LIMITATIONS: Based on the tests produced there are no limitations or shortfalls in the program.**

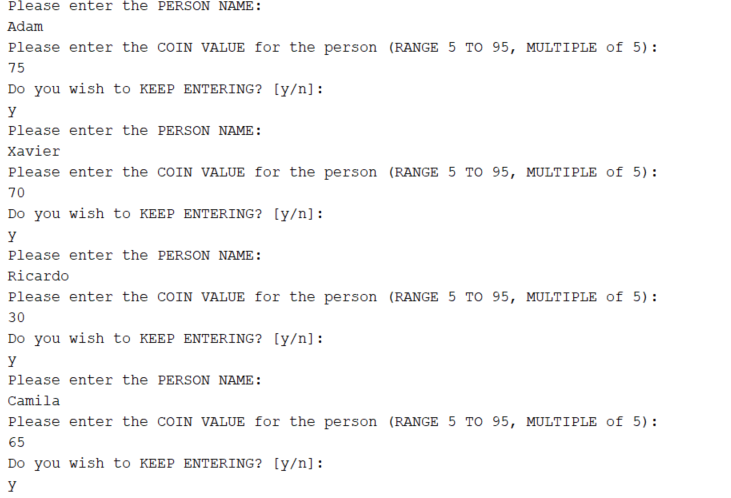
**6. TESTING / TABLE**

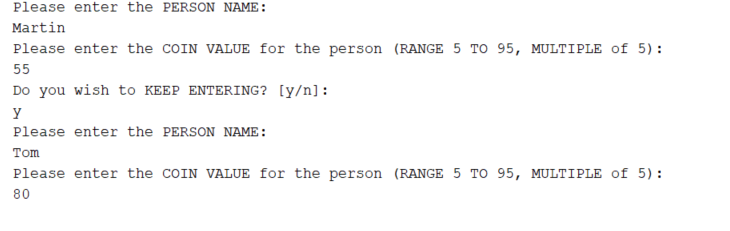
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TEST | DESCRIPTION | INPUT | EXPECTED OUTPUT | PASS/FAIL |
| 1 | VALID NUMBER OF PEOPLE | 12 | LOOP GETTING DATA 12X | PASS |
| 2 | MENU OPTION 1 | 1 AND RICARDO | Customer:  Ricardo 30 cents Change:  10 cents: 3 | PASS |
| 3 | MENU OPTION 1(NAME NOT IN THE  LIST) | 1 AND RONALDO | Name: RONALDO Not found | PASS |
| 4 | MENU OPTION 2 | 2 | The name with the SMALLEST amount is.: John with: 15 cents Change:  10 cents: 1  5 cents: 1 | PASS |
| 5 | MENU OPTION 3 | 3 | The name with the LARGEST amount is.: Fred with: 95 cents  Change:  50 cents: 1  10 cents: 4  5 cents: 1 | PASS |
| 6 | MENU OPTION 4 | 4 | The TOTAL OF FIFTY CENTS IS: 9  The TOTAL OF TEN CENTS IS: 22  The TOTAL OF FIVE CENTS IS: 7  The sum of all clients is: 38 | PASS |
| 7 | MENU OPTION 5 | 5 | You QUIT, thank you for using this program. Bye! | PASS |
| 8 | ANY INVALID MENU OPTION | 0 | The option you entered is INVALID TRY AGAIN. | PASS |
| 9 | NOT A MULTIPLE OF 5 | 3 | Please enter the COIN  VALUE for the person  (RANGE 5 TO 95,  MULTIPLE of 5): | PASS |

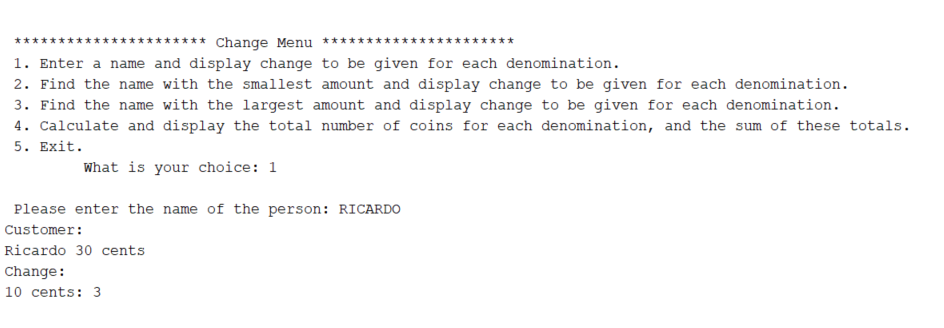
TEST1



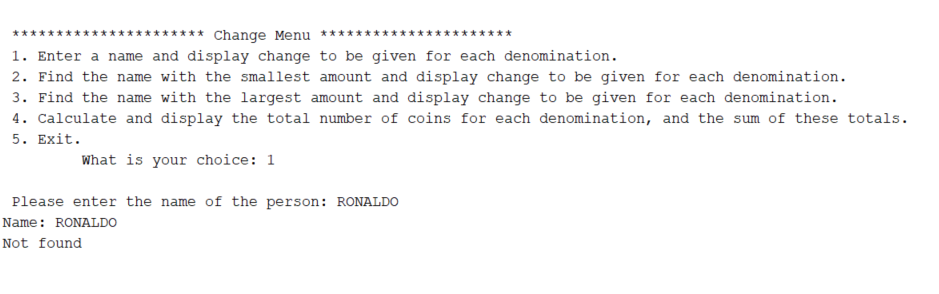




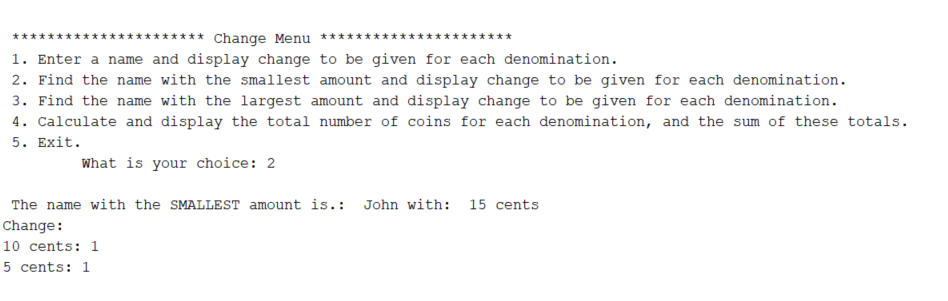


TEST 2

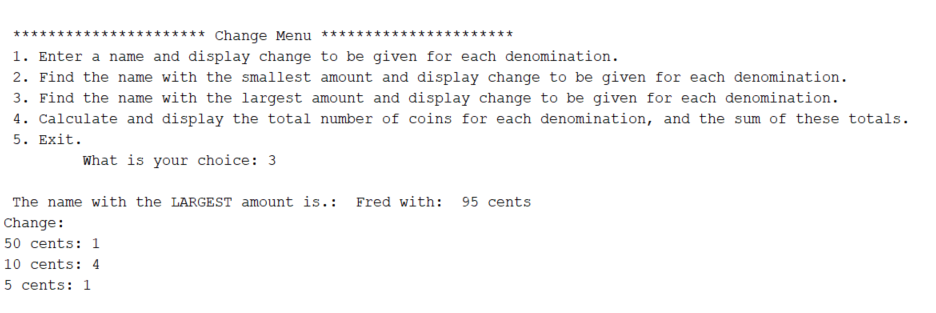
TEST 3



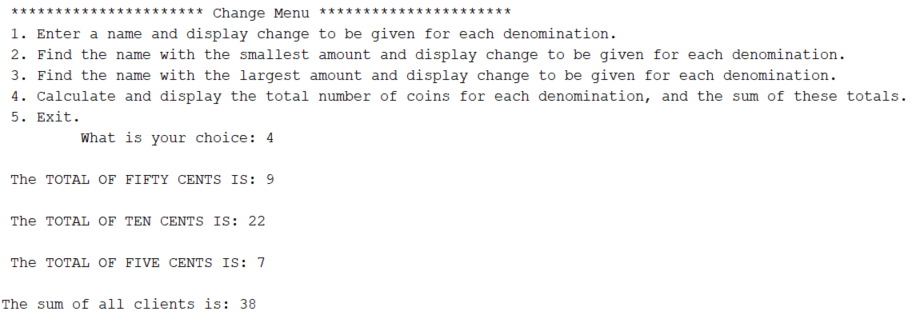
TEST 4



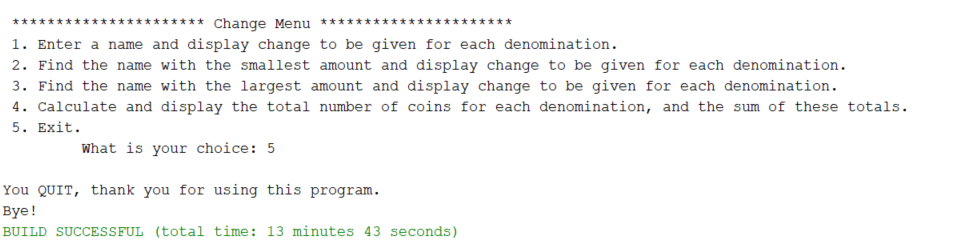
TEST 5



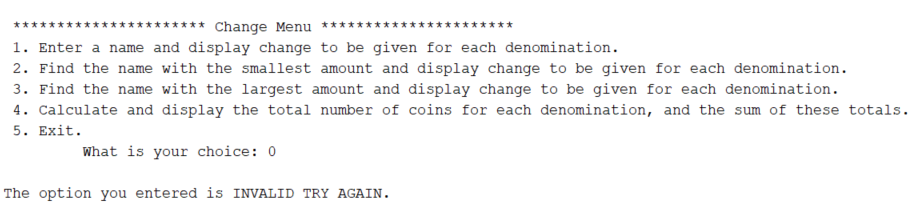
TEST 6



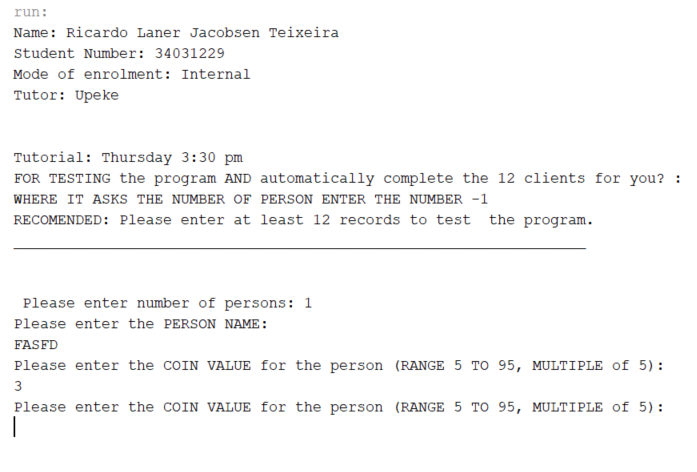
TEST 7



TEST 8



TEST 9



**4.SOURCE PROGRAM LISTING:**

**<CLIENT CLASS>**

/\*\*

\* Title: Client Change

\* @author Ricardo Laner Jacobsen Teixeira

\* Date: 01/09/2020

\* File name: Assignment

\* Purpose: This program was created with the intuition of helping the user on giving change based on a specific amount.

\* For testing purposes, it was added an initial statement when the program is running that gives instructions to auto input 12 customers on the array, so if the statement is followed, the program will run based on

12 clients.

\*

\*/

package assignment;

import java.util.InputMismatchException;

import java.util.Scanner;

public class Client {

public static Scanner keyboard = new Scanner (System.in);

//to control the position of the object on the array in the client and Change class as well.

static int c=0;

public static void main(String[] args) {

String name="";

//To receive the client amount

int amount=0;

//To control if the program continues

boolean keep=true;

char autoInput=' ';

//To check the range.

boolean inRange=false;

//To define the quantity of data stored into the array

int howMany =0;

//Controll the menu case

int menu = 0;

StudentInfo();

//Displays the instructions

displayInitialStatements();

howMany= keyboard.nextInt();

keyboard.nextLine();

// Creates an object of the class Change using parameterized constructo and passing number of persons as parameter

Change person = new Change(howMany);

//looping while the user wishes to continue.

while(keep==true ){

//-1 means hardcode so, we do not need to enter manually

if(howMany==-1){

//parametrizing the counter as the amountber of hardcoded of 12 people

c=12;

break;}//end of if

System.out.println("Please enter the PERSON NAME: ");

name=keyboard.nextLine();

String provisoryName=" ";

for (int i = 0; i < name.length(); i++) {

if(name.charAt(i)==' '){

name = provisoryName;

break;}else{

provisoryName+=name.charAt(i);}

}

// Looping till get the amount in range.

while(inRange==false){

while (true) { // loops forever until break

try { // checks code for exceptions

System.out.println("Please enter the COIN VALUE for the person (RANGE 5 TO 95, MULTIPLE of 5): ");

amount=keyboard.nextInt();

break; // if no exceptions breaks out of loop

}

catch (InputMismatchException e) { // if an exception appears prints message below

System.err.println("Please enter a number! ");

keyboard.next();

continue; // continues to loop if exception is found

}

}

//Checking if the amount entered is in range.

if(amount>=5 && amount<=95 && amount%5==0){

inRange=true;

}else{

inRange=false;

} // end of if

}//end of range while

int ret = person.findPerson(name);

if(c==0){person.addChange( name, amount);}

else if(ret!=-1){

person.peep[ret].addCoinValue(amount);

c--;

}

else if(ret==-1){

person.addChange( name, amount);

}

//incrementing the position

c++;

//postion of the arra must be the same as the number of person entered

//otherwise error will occur

if(c==howMany){break;}

//Checking if the user wants to continue.

System.out.println("Do you wish to KEEP ENTERING? [y/n]: ");

keep = person.keepEntering(keyboard.next().toUpperCase().charAt(0));

keyboard.nextLine();

//Reseting variable for while loop

inRange = false;

}//end of keep while

//Dispalay the menu

displayMenu();

// Accepts and returns user choice

menu = keyboard.nextInt();

while(menu>5 || menu<1){

System.out.println("\nThe option you entered is INVALID TRY AGAIN.");

displayMenu();

menu = keyboard.nextInt();

}

while( menu>=1 ){

//menu as the number

switch(menu){

case 1:

// Enter the name of the person

System.out.print("\n Please enter the name of the person: ");

name = keyboard.next();

int personFound=person.findPerson(name);

if(personFound==-1){

System.out.println("Name: "+name+"\n" +

"Not found");

}

else{

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue());

System.out.println("Customer: \n" + person.peep[personFound].getPersonName() + " "+ person.peep[personFound].getCoinValue() + " cents");

System.out.println("Change: ");

if(person.peep[personFound].fiftyCent != 0)

System.out.println("50 cents: " +person.peep[personFound]. fiftyCent);

// Checks if number of tens is not zero then display the data

if(person.peep[personFound].tenCent != 0)

System.out.println("10 cents: " + person.peep[personFound].tenCent);

// Checks if number of five is not zero then display the data

if(person.peep[personFound].fiveCent != 0)

System.out.println("5 cents: " + person.peep[personFound].fiveCent);

}//end of if statement

break;

case 2:

personFound= person.smallestAmount();

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue());

System.out.println("\n The name with the SMALLEST amount is.: " + person.peep[personFound].getPersonName() + " with: "+ person.peep[personFound].getCoinValue() + " cents");

System.out.println("Change: ");

if(person.peep[personFound].fiftyCent != 0)

System.out.println("50 cents: " +person.peep[personFound]. fiftyCent);

// Checks if number of tens is not zero then display the data

if(person.peep[personFound].tenCent != 0)

System.out.println("10 cents: " + person.peep[personFound].tenCent);

// Checks if number of five is not zero then display the data

if(person.peep[personFound].fiveCent != 0)

System.out.println("5 cents: " + person.peep[personFound].fiveCent);

break;

case 3:

personFound= person.largestAmount();

person.peep[personFound].calculateChange(person.peep[personFound].getCoinValue());

System.out.println("\n The name with the LARGEST amount is.: " + person.peep[personFound].getPersonName() + " with: "+ person.peep[personFound].getCoinValue() + " cents");

System.out.println("Change: ");

if(person.peep[personFound].fiftyCent != 0)

System.out.println("50 cents: " +person.peep[personFound]. fiftyCent);

// Checks if number of tens is not zero then display the data

if(person.peep[personFound].tenCent != 0)

System.out.println("10 cents: " + person.peep[personFound].tenCent);

// Checks if number of five is not zero then display the data

if(person.peep[personFound].fiveCent != 0)

System.out.println("5 cents: " + person.peep[personFound].fiveCent);

break;

case 4:

person.showTotal();

System.out.println("\n The TOTAL OF FIFTY CENTS IS: " + person.totalFifty);

System.out.println("\n The TOTAL OF TEN CENTS IS: " +person.totalTen);

System.out.println("\n The TOTAL OF FIVE CENTS IS: " +person.totalFive);

System.out.println("\nThe sum of all clients is: " + (person.totalFifty+person.totalTen+person.totalFive));

//Restarting the variables to not double values if case 4 is called again.

person.totalFifty=0;

person.totalTen=0;

person.totalFive=0;

break;

case 5:

System.out.println("\nYou QUIT, thank you for using this program.\nBye!");

System.exit(0);

}//End of Switch

displayMenu();

menu = keyboard.nextInt();

while(menu>5 || menu<1){

System.out.println("\nThe option you entered is INVALID TRY AGAIN.");

displayMenu();

menu = keyboard.nextInt();

}

}//end while menu

}//end of main method

public static void displayInitialStatements(){//Method to output the intial statements

System.out.println("FOR TESTING the program AND automatically complete the 12 clients for you? : ");

System.out.println("WHERE IT ASKS THE NUMBER OF PERSON ENTER THE NUMBER -1");

//Displaying the recomendation

System.out.println("RECOMENDED: Please enter at least 12 records to test the program.");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

System.out.println("");

// Enter the number of persons

System.out.print("\n Please enter number of people: ");

}

public static void displayMenu(){ // Displays menu

System.out.print("\n\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Change Menu \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.print("\n 1. Enter a name and display change to be given for each denomination.");

System.out.print("\n 2. Find the name with the smallest amount and display change to be given for each denomination.");

System.out.print("\n 3. Find the name with the largest amount and display change to be given for each denomination.");

System.out.print("\n 4. Calculate and display the total number of coins for each denomination, and the sum of these totals.");

System.out.print("\n 5. Exit.");

System.out.print("\n\t What is your choice: ");

}

public static void StudentInfo(){

System.out.println("Name: Ricardo Laner Jacobsen Teixeira");

System.out.println("Student Number: 34031229");

System.out.println("Mode of enrolment: Internal ");

System.out.println("Tutor: Upeke");

System.out.println("\n\nTutorial: Thursday 3:30 pm");

}

}//end of class

**<CHANGE CLASS>**

/\*\*

\* Title: Client Change

\* @author Ricardo Laner Jacobsen Teixeira

\* Date: 01/09/2020

\* File name: Assignment

\* Purpose: This program was created with the intuition of helping the user on giving change based on a specific amount.

\* For testing purposes, it was added an initial statement when the program is running that gives instructions to auto input 12 customers on the array, so if the statement is followed, the program will run based on

12 clients.

\*

\*/

package assignment;

public class Change {

private String personName;

private int coinAmount;

int fiftyCent;

int tenCent;

int fiveCent;

int totalFifty ;

int totalTen ;

int totalFive ;

//Declaring an array of object

Change peep[];

//default constructor

Change(){

personName="Unknown";

coinAmount=0;

}

// Constructor setting the values to variables

Change(String name, int amount)

{

this.personName = name;

this.coinAmount = amount;

}// End of parameterized constructor

// Constructor for the array

Change(int size)

{

//returning -1 mean to hardcode

if (size==-1){size=12;

peep =new Change[size];

this.hardCode();}else{

peep =new Change[size];

}

}// End of constructor

//Check if the name is already into the array

// pre-condition receive the client name as a parameter

//post-condition returns the client position.

public int findPerson(String name){

int position=-1;

for (int i = 0; i < (Client.c); i++) {

if(peep[i].personName.equalsIgnoreCase(name)){

position= i;

}

}

return position;

}

// pre-condition: receive the client name and the amount as a parameter to be set

//post-condition: Creates a new client object with name and amount.

void addChange(String name, int amount)

{

peep[Client.c] = new Change(name, amount);

}

// pre-condition: receive the client name as a parameter to be set

//post-condition: sets the client object name.

public void setPersonName(String name){

this.personName=name;

}

// pre-condition: receive the amount as a parameter

//post-condition: Calculates the denominations amount

void calculateChange(int amount)

{

// Calculates number of fifties available in amount

this. fiftyCent = amount / 50;

// Calculates remaining amount

amount = amount % 50;

// Calculates number of tens available in amount

this.tenCent = amount / 10;

// Calculates remaining amount

amount = amount % 10;

// Calculates number of fives available in amount

this. fiveCent = amount / 5;

// Calculates remaining amount

amount = amount % 5;

}// End of method

// pre-condition does not receive any parameter

//post-condition returns the client position with the smallest amount.

public int smallestAmount()

{

// Stores the first client amount as smallest amount

int smallest = peep[0].coinAmount;

// Stores the first client position as smallest position

int position = 0;

// Loops from 1 to end of the array of clients

for(int c = 1; c < Client.c; c++)

{

// Checks if current client amount is less than the earlier smallest amount

if(peep[c].coinAmount < smallest)

{

// Assigns the current client amount as the smallest amount

smallest = peep[c].coinAmount;

// Stores the current position as the smallest position

position = c;

}// End of if condition

}// End of for loop

return position;

}// End of method

// Method to find the client having largest amount and display

// client information with change

// pre-condition does not receive any parameter

//post-condition returns the client position with the largest amount.

public int largestAmount()

{

// Stores the first client amount as largest amount

int largest = peep[0].coinAmount;

// Stores the first client position as largest position

int position = 0;

// Loops from 1 to end of the array of clients

for(int c = 1; c < Client.c; c++)

{

// Checks if current client amount is greater than the earlier largest amount

if(peep[c].coinAmount > largest)

{

// Assigns the current client amount as the largest amount

largest = peep[c].coinAmount;

// Stores the current position as the largest position

position = c;

}// End of if condition

}// End of for loop

return position;

}// End of method

// pre-condition does not receive any parameter

//post-condition: Stores the total number of each denomination

public void showTotal(){

for(int i=0; i<Client.c; i++){

peep[i].calculateChange(peep[i].getCoinValue());

totalFifty+=peep[i].fiftyCent ;

totalTen+=peep[i].tenCent ;

totalFive+=peep[i].fiveCent ;

}

}

// pre-condition: receive the client amount as a parameter to be set

//post-condition: sets the client object coin amount.

public void setCoinValue(int coin){

this.coinAmount=coin;

}

// In case of duplicates in the array.

// pre-condition: receive the client amount as a parameter to be added

//post-condition: Adds the client object amount.

public void addCoinValue(int coin){

this.coinAmount+=coin;

}

//Returns the client name

public String getPersonName(){

return this.personName;

}

// return the client amount

public int getCoinValue(){

return this.coinAmount;

}

// to check if the user want to continue

public static boolean keepEntering (char resp){

boolean result=false;

if (resp=='Y'){

result = true;

}

else if (resp=='N'){

result = false;

}

return result;

}

public void hardCode(){

peep[0] = new Change("Jane",20);

peep[1] = new Change("John",15);

peep[2] = new Change("Fred", 95);

peep[3] = new Change("Wilma", 85);

peep[4] = new Change("Paul", 65);

peep[5] = new Change("Marry",50);

peep[6] = new Change("Adam",75);

peep[7] = new Change("Xavier",70);

peep[8] = new Change("Ricardo", 30);

peep[9] = new Change("Camila", 65);

peep[10] = new Change("Martin",55);

peep[11] = new Change("Tom", 80);

}

}//class end