#include <stdio.h>

#include <stdlib.h>

//Franklyn Gonzalez, last edited 1/22

double lowestDeposit(double deposit[]) {

int i;

int pass;

double temp;

for (i = 0; i < 10; i++)

printf("\n%.2lf", deposit[i]);

printf("\n");

for (pass = 1; pass <= 10 - 1; pass++) {

for (i = 0; i < 10 - 1; i++) {

if (deposit[i] > deposit[i + 1]) {

temp = deposit[i];

deposit[i] = deposit[i + 1];

deposit[i + 1] = temp;

}

}

}

printf("The lowest deposit that was made is $%.2lf", deposit[0]);

//return the lowest deposit from the bubble sort

return 0;

}

double averageDeposit(double deposit[]) {

int n = 0;

double sum = 0;

for (int i = 0; i < 10; i++) {

sum += deposit[ i ]; // set element at location i to 0

if (deposit[ i ] > 0)

n++;

}

//divide sum by number of values made, as the deposit

return (sum / n);

}

double allDeposits(double deposit[]) {

//bubble sort

int i, pass;

double temp;

for (i = 0; i < 10; i++)

printf("\n%.2lf ", deposit[i]);

printf("\n");

for (pass = 1; pass <= 10 - 1; pass++) {

for (i = 0; i < 10 - 1; i++) {

if (deposit[i] > deposit[i + 1]) {

temp = deposit[i];

deposit[i] = deposit[i + 1];

deposit[i + 1] = temp;

}

}

}

for (i = 0; i < 10; i++)

printf("\n%.2lf ", deposit[i]);

return 0;

}

double addDeposits(double deposit[]) {

double sum = 0;

for ( int i = 0; i < 10; ++i )

sum += deposit[ i ]; // set element at location i to 0

//return total sum

return sum;

}

double makeDeposit(double deposit[], int d) {

//Attempted to add ".2lf" but an error occurred, could not troubleshoot

scanf\_s("%lf", &deposit[d]);

printf("Processing deposit..");

//return deposit made

return deposit[d];

}

main()

{

//array used for deposits

double deposit[ 10 ] = { 0.0 };

//

int d = 0;

// 'm' used as a counter, for each value stored in the memory

int m = 0;

// 'x' used for switch

char x;

//defines array from the string

char bank[] = "MSDALQ";

//initalize array values

for ( int i = 0; i < 10; ++i ) {

deposit[ i ] = 0; // set element at location i to 0

do

{

//each menu letter represents a character from the array "bank[]"

printf("\nWelcome to the World Bank! Our maximum deposits are 10. \nEnter the options below.");

printf("\n[%c]ake a new deposit\n", bank[0]);

printf("[%c]um of all deposits\n", bank[1]);

printf("[%c]eposits will be displayed from the lowest to highest deposit\n", bank[2]);

printf("[%c]verage of all deposits\n", bank[3]);

printf("[%c]owest deposit will be displayed\n", bank[4]);

printf("[%c]uit the program\n\n", bank[5]);

//used as a buffer, experienced errors with getchar() repeating the menu options.

while ((getchar()) != '\n');

//requests char input

x = getchar();

//selects from menu

switch (x)

{

case 'M':

//Letter "M" from x

printf("\nMaking a new deposit.... ");

//calls makeDeposit and stores value in memory '[m++]'

printf("\nYou have made $%.2lf as your initial deposit.\n", makeDeposit(deposit, m++));

break;

case 'S':

//Letter "S" from x

printf("\nAdding up all deposits....");

//calls addDeposits

printf("$%.2lf\n" , addDeposits(deposit));

break;

case 'D':

//Letter "D" from x

printf("\nDeposits will be displayed from the lowest to highest deposit.....");

//calls allDeposits

allDeposits(deposit);

break;

case 'A':

//Letter "A" from x

printf("\nAveraging all deposits.....");

//calls averageDeposit

printf("\nThe average of all deposits is $%.2lf \n", averageDeposit(deposit));

break;

case 'L':

//Letter "L" from x

printf("\nLowest deposit will be displayed.....");

//calls lowestDeposit

lowestDeposit(deposit);

break;

case 'Q':

//Letter "Q" from x

printf("\nQuitting program.\n");

break;

default:

//No characters are received from input, thus falling back on the break

printf("\nInvalid response. Returning back to the main menu.\n");

break;

}

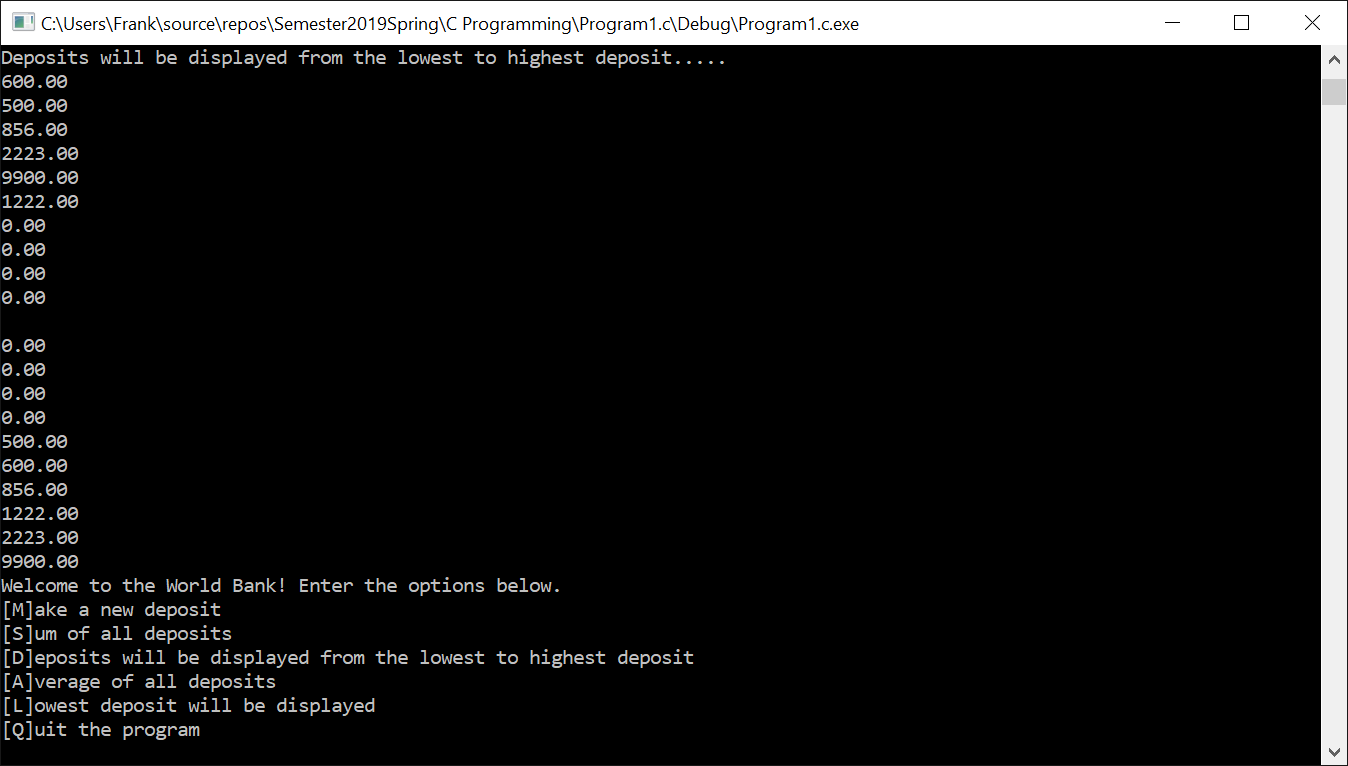
} while (x == 'M' || x == 'S' || x == 'D' || x == 'A' || x == 'L' || x != 'Q');

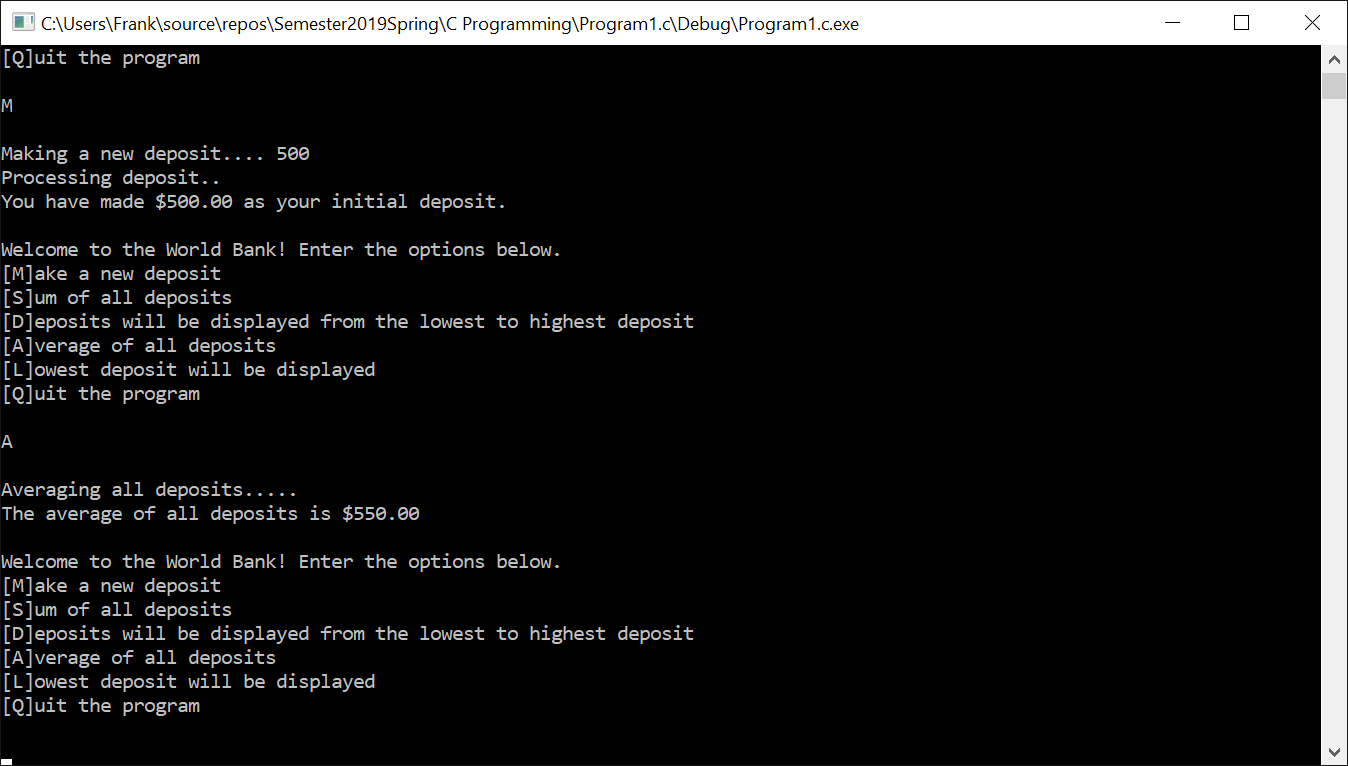
//return value used to end program

return 0;

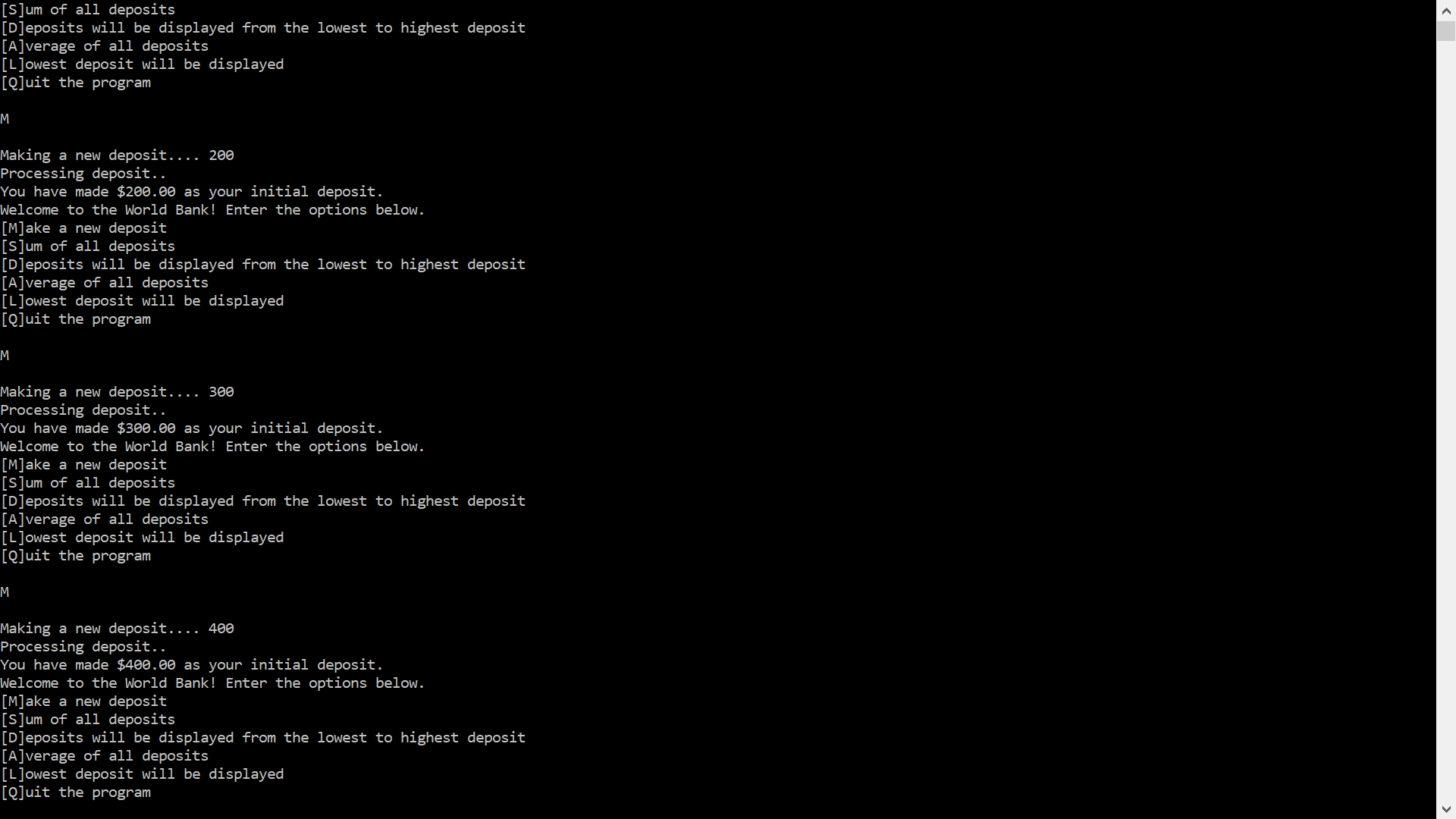
}

}









**IPO**

|  |  |  |
| --- | --- | --- |
| **Input**  - deposit[ 10 ],  - d  - m  - x  - bank[x]  - deposit[ i ]  - makeDeposit()  - addDeposits()  - allDeposits()  - averageDeposit()  - lowestDeposit()  - deposit[d]  - sum  - pass  - i  - temp  - n | **Processing**  Initalize variables and values inside array. Identify array string with characters inside bank[x].  **Do** Print menu. **Enter** one of the options below:   1. Deposit 2. Add deposits 3. Display all deposits (from lowest to highest) 4. Average deposits (that were made) 5. Show the lowest deposit 6. QUIT   **While** button choices are true, loop continues to run program. If **user** provides invalid option, the person returns to the main menu.  **User** acknowledges options and identifies each value by the option that was set in the main menu.  i.e. Average displays average of deposits. | **Output**  Output a deposit by option **M**  Output the sum of all deposits by option **S**  Output all the deposits stored in the memory of the array by option **D**  Output the average of the deposits by option **A**  Output the lowest deposit by option **L**  Output an invalid option.  Output the quit program by option **Q** |