/\*

\* File Name: A6Calculator.cpp

\* Names: Anthony Garcia

\* Course: COP 1000C at Valencia College

\* Professor: David Stendel

\* Description: This program displays a menu that loops

\* to the user with the following options and their related operations:

\* Option Operation

\* (G)et two numbers. Prompts the user for two numbers.

\* (A)dd. Adds the three numbers.

\* (S)ubtract. Subtracts the first number from the second.

\* (M)ultiply. Multiplies the numbers.

\* (D)ivide. Divides the first number by the second.

\* a(V)erage. Finds the average of the two numbers

\* (L)argest. Finds the largest of the two numbers.

\* (P)ositive, negative, or zero. Tells the user if the numbers are

\* positive, negative, or zero.

\* e(X)it. Exits the program without

\* executing an operation.

\* Thank You and Goodbye Message: A thank you and goodbye message is

\* displayed at the end and when the user exits the program.

\* Error Messages: An error message will display in 3 instances.

\* 1. When the user enters an invalid choice.

\* 2. When the user wishes to divide but the second and third numbers are zeroes.

\* 3. When they try selecting an option before selecting option G.

\* Date: 11/19/21

\*/

// Preprocessor Directives

#include <iostream>

#include <iomanip>

using namespace std;

// Prototypes

double addNumbers(double NUMBERS[], int capacity, double answer);

double averageNumber(double NUMBERS[], int capacity, double answer);

void displayMenu();

double divideNumbers(double NUMBERS[], int capacity, double answer);

double LargestNumber(double NUMBERS[], int capacity, double answer);

double posNegZero(double NUMBERS[], int capacity, double answer);

bool processMenuChoice(double NUMBERS[], int capacity, bool numbersGotten, double answer);

double subtractNumbers(double NUMBERS[], int capacity, double answer);

double userInputs(double NUMBERS[], int capacity);

/\*

\* Name: main()

\* Parameters: None.

\* Processes: This function calls other functions to display a menu, perform calculations,

\* and process the user's menu selections.

\* Return Value: An integer representing an error code; if the program ends without error,

\* zero will be returned to the calling program or operating system.

\*/

int main()

{

// Constants and Variables

char choice = '\0';

double answer = 0.0;

bool numbersGotten = false;

const int SIZE = 3;

double NUMBERS[SIZE] = { 0.0, 0.0, 0.0};

// Functions

void displayMenu();

bool processMenuChoice(double NUMBERS[], int SIZE, bool numbersGotten, double answer);

// Calculator Processing

do

{

// Input - Menu

displayMenu();

// Menu Choice Processor

numbersGotten = processMenuChoice(NUMBERS, SIZE, numbersGotten, answer);

}

while (choice != 'x' && choice != 'X');

return 0;

}

/\*

\* Name: addNumbers()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: This function adds the three numbers.

\* Return Value: answer The sum.

\*/

double addNumbers(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

for (int i = 1; i < capacity; i++)

{

answer += NUMBERS[i];

return answer;

}

}

/\*

\* Name: averageNumber()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: This function finds the average of the two numbers

\* Return Value: answer The average.

\*/

double averageNumber(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

for (int i = 1; i < capacity; i++)

{

answer += NUMBERS[i];

}

return answer = answer / capacity;

}

/\*

\* Name: DisplayMenu()

\* Parameters: None.

\* Processes: This function displays the menu.

\* Return Value: None.

\*/

void displayMenu()

{

cout << "\nCalculator Menu\n\n";

cout << "(G)et two numbers.\n";

cout << "(A)dd.\n";

cout << "(S)ubtract.\n";

cout << "(M)ultiply.\n";

cout << "(D)ivide.\n";

cout << "a(V)erage.\n";

cout << "(L)argest.\n";

cout << "(P)ositive, negative, or zero.\n";

cout << "e(X)it.\n\n";

cout << "Choice: ";

}

/\*

\* Name: divideNumbers()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: This function divides the first number by the second.

\* Return Value: answer The quotient.

\*/

double divideNumbers(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

for (int i = 1; i < capacity; i++)

{

return answer /= NUMBERS[i];

}

}

/\*

\* Name: largestNumber()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer where the answer is stored.

\* Processes: This function finds the largest of the two numbers.

\* Return Value: answer The largest number.

\*/

double LargestNumber(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

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

{

if (NUMBERS[i] > answer)

{

return answer = NUMBERS[i];

}

}

}

/\*

\* Name: multiplyNumbers()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: This function Multiplies the numbers.

\* Return Value: answer The product.

\*/

double multiplyNumbers(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

for (int i = 1; i < capacity; i++)

{

return answer \*= NUMBERS[i];

}

}

/\*

\* Name: posNegZero()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: This function Tells the user if the numbers are positive, negative, or zero.

\* Return Value: answer Identifies each number as positive, negative, or zero.

\*/

double posNegZero(double NUMBERS[], int capacity, double answer)

{

// Calculation Function

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

{

// Determination

if (NUMBERS[i] > 0.0)

{

// Output - Positive

cout << "\nThe number " << NUMBERS[i] << " is positive.\n";

}

else

{

if (NUMBERS[i] < 0.0)

{

// Output - Negative

cout << "\nThe number " << NUMBERS[i] << " is negative.\n";

}

else

{

// Output - Zero

cout << "\nThis number is zero.\n";

}

}

}

return 0;

}

/\*

\* Name: processMenuChoice()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* numbersGotten Tells the program if the user has inputted the numbers.

\* answer Where the answer is stored.

\* Processes: This function processes the majority of the menu and calls other functions to

\* perform calculations.

\* Return Value: numbersGotten Returns this variable as true or false.

\*/

bool processMenuChoice(double NUMBERS[], int capacity, bool numbersGotten, double answer)

{

// Output Formatting

cout << fixed << setprecision(3);

// Constants and Variables

char choice = '\0';

capacity = 3;

double userInputs(double NUMBERS[], int capacity);

double subtractNumbers(double NUMBERS[], int capacity, double answer);

double posNegZero(double NUMBERS[], int capacity, double answer);

double multiplyNumbers(double NUMBERS[], int capacity, double answer);

// Input Letter

cin >> choice;

// Menu Processing

switch (choice)

{

// Numbers

case 'G':

case 'g':

// Input

cout << "\nEnter three numbers separated with a space: ";

userInputs(NUMBERS, capacity);

numbersGotten = true;

break;

// Addition

case 'A':

case 'a':

// Verification

if (numbersGotten == true)

{

// Calculation Function

answer = addNumbers( NUMBERS, capacity, answer);

// Output

cout << "\nAnswer: ";

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

{

cout << NUMBERS[i];

if (i != capacity - 1)

{

cout << " + ";

}

}

cout << " = " << answer << "\n";

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Subtraction

case 'S':

case 's':

// Verification

if (numbersGotten == true)

{

// Calculation Function

answer = subtractNumbers(NUMBERS, capacity, answer);

// Output

cout << "\nAnswer: ";

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

{

cout << NUMBERS[i];

if (i != capacity - 1)

{

cout << " - ";

}

}

cout << " = " << answer << "\n";

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Multiplication

case 'M':

case 'm':

// Verification

if (numbersGotten == true)

{

// Calculation Function

answer = multiplyNumbers(NUMBERS, capacity, answer);

// Output

cout << "\nAnswer: ";

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

{

cout << NUMBERS[i];

if (i != capacity - 1)

{

cout << " \* ";

}

}

cout << " = " << answer << "\n";

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Division

case 'D':

case 'd':

// Verification

if (numbersGotten == true)

{

// Verification

if (NUMBERS[1] != 0.0 && NUMBERS[2] != 0.0)

{

// Calculation Function

answer = divideNumbers(NUMBERS, capacity, answer);

// Output

cout << "\nAnswer: ";

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

{

cout << NUMBERS[i];

if (i != capacity - 1)

{

cout << " / ";

}

}

cout << " = " << answer << "\n";

}

else

{

// Error

cout << "\n\tError: Cannot divide by zero!\n";

}

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Average

case 'V':

case 'v':

// Verification

if (numbersGotten == true)

{

// Calculation Function

answer = averageNumber(NUMBERS, capacity, answer);

// Output

cout << "\nAnswer: (";

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

{

cout << NUMBERS[i];

if (i != capacity - 1)

{

cout << " + ";

}

}

cout << ") / " << 3 << " = " << answer << "\n";

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Largest

case 'L':

case 'l':

// Verification

if (numbersGotten == true)

{

// Determination Function

answer = LargestNumber(NUMBERS, capacity, answer);

// Output

cout << "\nThe largest number is " << answer << ".\n";

}

else

{

// Error

cout << "\n\tError: Please select option G from the menu first!\n";

}

break;

// Positive, Negative, or Zero

case 'P':

case 'p':

// Verification

if (numbersGotten == true)

{

// Determination Function

answer = posNegZero(NUMBERS, capacity, answer);

}

else

{

// Error - Menu

cout << "\n\tError: Please select option G from the menu first!\n\n";

}

break;

// Exit

case 'X':

case 'x':

// Output - Good-bye Message

cout << "\nThank you for using Anthony's calculator! Good-bye!\n";

break;

default:

// Error

cout << "\n\tError: Invalid choice!\n";

}

return numbersGotten;

}

/\*

\* Name: subtractNumbers()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* answer Where the answer is stored.

\* Processes: Subtracts the first number from the second.

\* Return Value: answer The difference.

\*/

double subtractNumbers(double NUMBERS[], int capacity, double answer)

{

// Calculation

answer = NUMBERS[0];

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

{

return answer -= NUMBERS[i];

}

}

/\*

\* Name: userInputs()

\* Parameters: NUMBERS[] an array.

\* capacity The size of the array.

\* Processes: Stores user's inputted numbers into an array.

\* Return Value: An integer representing an error code; if the program ends without error, zero

\* will be returned to the calling program or operating system.

\*/

double userInputs(double NUMBERS[], int capacity)

{

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

{

cin >> NUMBERS[i];

}

return 0;

}