```
/*
Memo P5
Created by: Jaco du Toit
Date: 23 March 2022
*/

#include "libVect.h"
#include <iostream>

using namespace std;
using namespace VectorSpace;

int main()
{
    //Seeding the random number generator
    srand(time(0));
    //Declare the five vectors
    vector<int> vecData;
    vector<int> vecSmall;
    vector<int> vecLarge;
    vector<int> vecSame;
    vector<int> vecSum;

    //Basic variables used in the main loop
    bool blnContinue = true;
    char chInput = '\0';

    do
    {
        //Output the menu
        system("cls");
        OutputMenu();

        //Get input
        cin >> chInput;
        chInput = tolower(chInput);
        switch(chInput)
        {
        case 'a':
            OutputVectors(vecData,vecSmall,vecLarge,vecSame,vecSum);
            break;
        case 'b':
            ClearVectors(vecData,vecSmall,vecLarge,vecSame,vecSum);
            break;
        case 'c':
            vecData = FillVector();
            break;
        case 'd':
            ModifyVectors(vecData,vecSmall,vecLarge,vecSame);
            break;
        case 'e':
            vecSum = GetVecSum(vecData);
            break;
        case 'x':
            blnContinue = false;
            break;
        default:
            cerr << "Please select a valid option" << endl;
            Pause();
        }
    }while(blnContinue);
    return 0;
}
```

```cpp
#ifndef LIBVECT_H_INCLUDED
#define LIBVECT_H_INCLUDED

#include <iostream>
#include <cstdlib>
#include <ctime>
#include <vector>
#include <cctype>

using namespace std;

namespace VectorSpace
{
    void OutputMenu();
    void Pause();
    void OutputVectors(vector<int> vecData, vector<int> vecSmall, vector<int> vecLarge, vector<int> vecSame, vector<int> vecSum);
    void ClearVectors(vector<int>& vecData, vector<int>& vecSmall, vector<int>& vecLarge, vector<int>& vecSame, vector<int>& vecSum);
    vector<int> FillVector();
    void ModifyVectors(vector<int> vecData, vector<int>& vecSmall, vector<int>& vecLarge, vector<int>& vecSame);
    vector<int> GetVecSum(vector<int> vecData);
}

#endif // LIBVECT_H_INCLUDED

```

```cpp
#include "libVect.h"

namespace VectorSpace
{
    //Generates a random number between two numbers
    int GetRand(int intLower, int intUpper)
    {
        int intRange = intUpper - intLower + 1;
        return rand()%intRange + intLower;
    }

    //Menu options
    void OutputMenu()
    {
        cout << "Please select one of the following options:" << endl;
        cout << "A: Display each of the five vectors" << endl;
        cout << "B: Empties each of the five vectors" << endl;
        cout << "C: Fills vecData with number of random numbers" << endl;
        cout << "D: Getting a number to modify smaller, larger and same vectors" << endl;
        cout << "E: Calculates the vecSum vector" << endl;
        cout << "X: Ends the program" << endl;
    }

    //Pauses the program
    void Pause()
    {
        cin.ignore(100,'\n');
        cout << "Press Enter to continue" << endl;
        cin.get();
    }

    //Outputs the entries in an integer vector
    void OutputVector(vector<int> vecNums)
    {
        for(int n:vecNums)
        {
            cout << n << " ";
        }
        cout << endl;
    }

    //Outputs each of the vectors
    void OutputVectors(vector<int> vecData, vector<int> vecSmall, vector<int> vecLarge, vector<int> vecSame, vector<int> vecSum)
    {
        cout << "vecData:" << endl;
        OutputVector(vecData);
        cout << "vecSmall:" << endl;
        OutputVector(vecSmall);
        cout << "vecLarge:" << endl;
        OutputVector(vecLarge);
        cout << "vecSame:" << endl;
        OutputVector(vecSame);
        cout << "vecSum:" << endl;
        OutputVector(vecSum);
        Pause();
    }

    //Clears each of the vectors
    void ClearVectors(vector<int>& vecData, vector<int>& vecSmall, vector<int>& vecLarge, vector<int>& vecSame, vector<int>& vecSum)
    {
        vecData.clear();
        vecSmall.clear();
        vecLarge.clear();
        vecSame.clear();
        vecSum.clear();
    }

    //Returns an integer number
    int GetInt()
    {
        int intNum;
        cin >> intNum;
        while(cin.fail())
        {
            cin.clear();
            string strJunk;
            cin >> strJunk;
            cout << "Please provide a proper number:";
            cin >> intNum;
        }
        return intNum;
    }
    //Returns an integer number that is at least a minimum of the given argument
    int GetMinimum(int intMin)
    {
        int intNum;
        do
        {
            cout << "Size of the number (>" << intMin << "):";
            intNum = GetInt();
        }while(intNum<intMin);
        return intNum;
    }

    //Creates a vector and fills it with user supplied random numbers in a user defined range
    vector<int> FillVector()
    {
        vector<int> vecNums;
        int intLower = 0;
        int intUpper = 0;
        int intCount = 0;

        cout << "Please provide the number of items in the vector:";
        intCount = GetMinimum(1);

        cout << "Please provide the lowest random number:";
        intLower = GetInt();

        cout << "Please provide the highest random number:";
        intUpper = GetMinimum(intLower+1);

        for(int n=0;n<intCount;n++)
        {
            vecNums.push_back(GetRand(intLower,intUpper));
        }
        return vecNums;
    }

    //Modify the vecSmall, vecLarge and vecSame vectors
    void ModifyVectors(vector<int> vecData, vector<int>& vecSmall, vector<int>& vecLarge, vector<int>& vecSame)
    {
        int intValue;
        cout << "Please provide the number that defines the vector split:";
        intValue = GetInt();
        vecSmall.clear();
        vecLarge.clear();
        vecSame.clear();
        for(int n:vecData)
        {
            if(n==intValue)
                vecSame.push_back(n);
            else if(n<intValue)
                vecSmall.push_back(n);
            else
                vecLarge.push_back(n);
        }
    }

    int CountDigits(int intVal)
    {
        //First approach: -15 = -1 + 5 = 4   Second approach:-15 = -1 + (-5) = -6
        //This solution uses the first approach but the second approach is also valid.
        bool blnNegative = false;           //Flag to check if we worked with a negative number
        //Determine if the number is negative. If so, remember, but change to positive for now
        if(intVal<0)
        {
            intVal = intVal * -1;
            blnNegative = true;
        }
        int intSum = 0;                     //The sum of the digits
        int intRem = 0;                     //The remainder after modulus 10.
        do
        {
            //If we are busy with the digits and the digits are less than 10, then handle the negative case specifically.
            if(intVal<10)
            {
                //If the number was negative, then transform the last digit back into a negative number
                if(blnNegative)
                    intVal = intVal *-1;
            }
            intRem = intVal % 10;           //Returns the least significant digit in the number
            intSum+=intRem;                 //Adds the least significant digit to the sum.
```

```
intSum = intSum + intRem
            //cout << intRem << endl;
            intVal = intVal / 10;           //Removes the least significant digit
            //cout << "intVal:" << intVal << endl;
        }while(intVal!=0);                  //Repeat until the value is zero

        return intSum;
    }

    //Calculates the sum of the digits for each of the numbers in the vecNum vector.
    vector<int> GetVecSum(vector<int> vecData)
    {
        vector<int> vecSum;
        for(int n:vecData)
        {
            vecSum.push_back(CountDigits(n));
        }
        return vecSum;
    }
}
```