CS205 C/C++ Programming - Project_1

Name: 刘乐奇（Liu Leqi）

SID： 12011327

Part 1 - Analysis

> It is easy to achieve positive integers multiplication by using the operator * in cpp. However, there are some other situation we need to think of.

Some users may want to calculate the multiplication of negative integers, and the other may want to calculate the multiplication of large integers. Besides, some users would probably mistouch the keyboard so that they might enter some characters that are not integers.

I think it safe to use `string` to incept the inputs so that the program would not breakdown as user enter illegal inputs.
We need to detect if users enter the right input first, which must be integers with not zero front. After that, we need to detect whether there are two negative integers or not, and then to decide whehter the answer cantains the minus. The `string` type variables may be use in case users want to calculate large integers.

In this project, we can use `iostream` for input and output, `string` for string type variables.

After all, if users want to calculate for more than once, the design of loop can make it.

Part 2 - Code

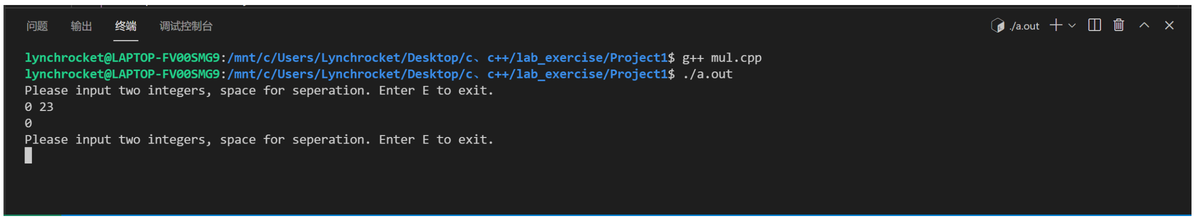
```
#include<iostream>
#include<string>

#define ll long long
#define ull unsigned long long

using namespace std;

inline bool legalNumber(string s);
inline bool isPositive(string s);
inline string Clear(string s);
inline string Multiply(string a, string b);

int main()
{
    ios::sync_with_stdio(0),cin.tie(0),cout.tie(0);

    string a = "", b = "";
    while(a != "E"){
        string result = "";

        cout << "Please input two integers, space for seperation. Enter E to
exit." << endl;
        input: cin >> a;
        if(a == "E") break;
        cin >> b;
        //first check whether the two inputs are number
        if(!legalNumber(a) | !legalNumber(b)) {
            cout << "The input is illegal! Please try again or E to exit." <<
endl;
            goto input;
        }

        //second check whether the two number is positive or negative
        if(isPositive(a) ^ isPositive(b)){//xor, when only one of a and b is
negative, true and get in.
            result += "-";
        }

        //third delete the front zero and minus
        a = Clear(a);
        b = Clear(b);

        //forth multiply
        if(a == "0" | b == "0") result += "0";
        else result += Multiply(a,b);

        cout << result << endl;
    }

    cout << "Thank you for calculating. See you next time." <<endl;
    return 0;
}

bool legalNumber(string s)
{
    if(s[0] == '-' | s[0] == '+'){
        s = s.substr(1);
    }
    int che = 0;
    for(int i = 0; i < s.size(); i++){
        if(s[i] >= '0' & s[i] <= '9'){
            che++;
        }
    }
    if(che == s.length()) return true;
    else return false;
}

bool isPositive(string s)
{
    if(s[0]=='-'){
        return false;
    }
    return true;
}

string Clear(string s)
{
    int i = 0;
    for( ; i < s.size(); i++){
        if(s[i] != '0' & s[i] != '-' & s[i] != '+'){
            break;
        }
    }
    if(s.substr(i).length() == 0) return s;
    else return s.substr(i);
}

//multiplication for large numbers, in hand calculation way
string Multiply(string a, string b)
{
    string result = "";
    int al[1010] = {0};
    int bl[1010] = {0};
    int tmp[2020] = {0};
    int i = 0, j = 0;

    for(i = a.length() - 1, j = 0; i >= 0; i--, j++){
        al[j] = a[i] - '0';
    }
    for(i = b.length() - 1, j = 0; i >= 0; i--, j++){
        bl[j] = b[i] - '0';
    }

    //calculate
    for(i = 0; i < a.length(); i++){
        for(j = 0; j < b.length(); j++){
            tmp[i + j] += al[i] * bl[j];
        }
    }

    //carry
    for(i = 0; i < (a.length() + b.length()); i++){
        if(tmp[i] > 9){
            tmp[i+1] += tmp[i]/10;
            tmp[i] %= 10;
        }
    }

    for(i = a.length() + b.length(); i >= 0 ; i--){
        if(tmp[i] == 0) continue;
        else break;
    }

    for(; i >=0; i--) result += to_string(tmp[i]);
    return result;
}
```

Part 3 - Result & Verification

> In this part, I will present some instance of my program output. Some are normal, some may be seen to be ridiculous in case the unexpected situation.

Test case #1:

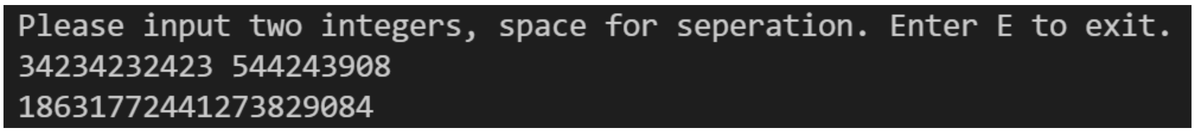
```
Input: 0 23
output: 0
```



```
lynchrocket@LAPTOP-FV00SMG9:/mnt/c/Users/Lynchrocket/Desktop/c、c++/lab_exercise/Project1$ g++ mul.cpp
lynchrocket@LAPTOP-FV00SMG9:/mnt/c/Users/Lynchrocket/Desktop/c、c++/lab_exercise/Project1$ ./a.out
Please input two integers, space for seperation. Enter E to exit.
0 23
0
Please input two integers, space for seperation. Enter E to exit.
```

Test case #2:

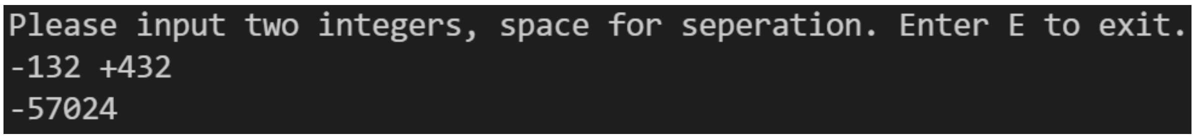
```
Input: 34234232423 544243908
output: 18631772441273829084
```



```
Please input two integers, space for seperation. Enter E to exit.
34234232423 544243908
18631772441273829084
```

Test case #3:

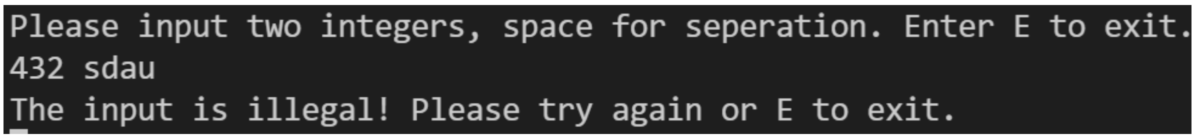
```
Input: -132 +432
output: -57024
```



```
Please input two integers, space for seperation. Enter E to exit.
-132 +432
-57024
```

Test case #4:

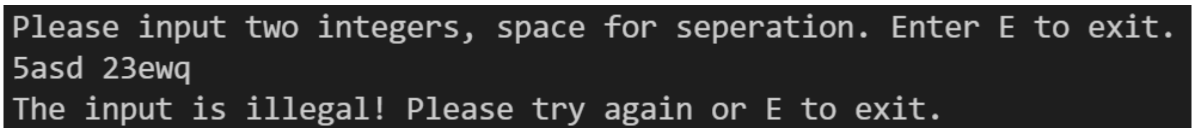
```
Input: 432 sdau
output: The input is illegal! Please try again or E to exit.
```



```
Please input two integers, space for seperation. Enter E to exit.
432 sdau
The input is illegal! Please try again or E to exit.
```

Test case #5:

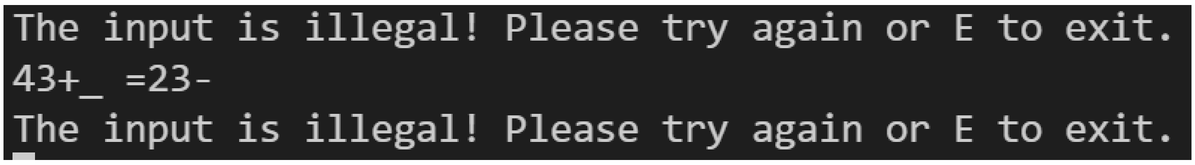
```
Input: 5asd 23ewq
output: The input is illegal! Please try again or E to exit.
```



```
Please input two integers, space for seperation. Enter E to exit.
5asd 23ewq
The input is illegal! Please try again or E to exit.
```

Test case #6:

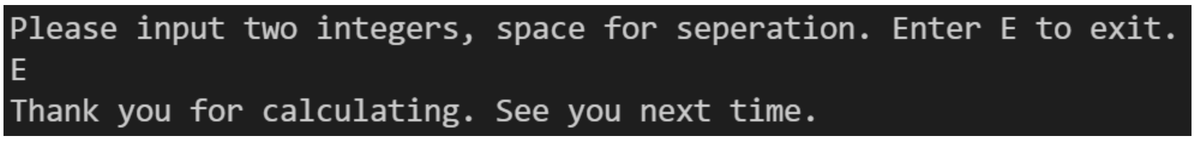
```
Input: 43+_ =23-
output: The input is illegal! Please try again or E to exit.
```



```
The input is illegal! Please try again or E to exit.
43+_ =23-
The input is illegal! Please try again or E to exit.
```

Test case #7 (Exit in common):

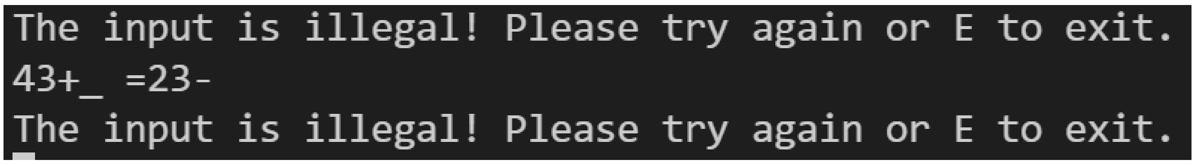
```
Input: E
output: Thank you for calculating. See you next time.
```



```
Please input two integers, space for seperation. Enter E to exit.
E
Thank you for calculating. See you next time.
```

Test case #8 (Exit after illegal input):

```
Input: E
output: Thank you for calculating. See you next time.
```



```
The input is illegal! Please try again or E to exit.
43+_ =23-
The input is illegal! Please try again or E to exit.
```

Part 4 - Difficulties & Solutions

- Originally, my program is very crude. It was written as soon as I saw the project.

```
#include<iostream>
using namespace std;
int main()
{
    int a = 0, b = 0;
    cout << "Please input two integers" << endl;
    cin >> a >> b;
    cout << a * b << endl;
    return 0;
}
```

I had no sooner found that it is too fragile than I saw the exmples in the description. So I optimized it.

- In case of the input is negative or positive but with plus in the front, or maybe large numbers so that are over the range of Integer type (even using `long long` which I had once thought of), or some illegal input with characters, I desided to use `string` to store the input and the output.

```
#include<iostream>
#include<string>

#define ll long long
```

```
#define ull unsigned long long

using namespace std;

inline bool legalNumber(string s);
inline bool isPositive(string s);
inline string Clear(string s);
inline string Multiply(string a, string b);

int main()
{
    string a = "", b = "";
    string result = "";

    cout << "Please input two integers, space for seperation." << endl;
    cin >> a >> b;

    //first check whether the two inputs are number
    if(!legalNumber(a) | !legalNumber(b)) {
        cout << "The input is illegal! " << endl;
    }

    //second check whether the two number is positive or negative
    if(isPositive(a) ^ isPositive(b)){//xor, when only one of a and b is
negative, true and get in.
        result += "-";
    }

    //third delete the front zero and minus
    a = Clear(a);
    b = Clear(b);

    //forth multiply
    result += Multiply(a,b);

    cout << result << endl;
    return 0;
}

bool legalNumber(string s)
{
    if(s[0] == '-' | s[0] == '+'){
        s = s.substr(1);
    }
    int che = 0;
    for(int i = 0; i < s.size(); i++){
        if(s[i] >= '0' & s[i] <= '9'){
            che++;
        }
    }
    if(che == s.length()) return true;
    else return false;
}

bool isPositive(string s)
{
    if(s[0]=='-'){
        return false;
    }
    return true;
}

string Clear(string s)
{
    int i = 0;
    for( ; i < s.size(); i++){
        if(s[i] != '0' & s[i] != '-' & s[i] != '+'){
            break;
        }
    }
    if(s.substr(i).length() == 0) return s;
    else return s.substr(i);
}

//multiplication for large numbers, in hand calculation way
string Multiply(string a, string b)
{
    string result = "";
    int al[1010] = {0};
    int bl[1010] = {0};
    int tmp[2020] = {0};
    int i = 0, j = 0;

    for(i = a.length() - 1, j = 0; i >= 0; i--, j++){
        al[j] = a[i] - '0';
    }
    for(i = b.length() - 1, j = 0; i >= 0; i--, j++){
        bl[j] = b[i] - '0';
    }

    //calculate
    for(i = 0; i < a.length(); i++){
        for(j = 0; j < b.length(); j++){
            tmp[i + j] += al[i] * bl[j];
        }
    }

    //carry
    for(i = 0; i < (a.length() + b.length()); i++){
        if(tmp[i] > 9){
            tmp[i+1] += tmp[i]/10;
            tmp[i] %= 10;
        }
    }

    for(i = a.length() + b.length(); i >= 0 ; i--){
        if(tmp[i] == 0) continue;
        else break;
    }

    for(; i >=0; i--) result += to_string(tmp[i]);
    return result;
}
```

- After all, I wanted to calculate more than once but excute only once, so I added `while()` and optimized the `main()` function. (the code posted below only contains `main()` function)

```
int main()
{
    string a = "", b = "";
    while(a != "E"){
        string result = "";

        cout << "Please input two integers, space for seperation. Enter E to 
exit." << endl;
        input: cin >> a;
        if(a == "E") break;
        cin >> b;

        //first check whether the two inputs are number
        if(!legalNumber(a) | !legalNumber(b)) {
            cout << "The input is illegal! Please try again or E to exit." << 
endl;
            goto input;
        }

        //second check whether the two number is positive or negative
        if(isPositive(a) ^ isPositive(b)){//xor, when only one of a and b is 
negative, true and get in.
            result += "-";
        }

        //third delete the front zero and minus
        a = Clear(a);
        b = Clear(b);

        //forth multiply
        result += Multiply(a,b);

        cout << result << endl;
    }

    cout << "Thank you for calculating. See you next time." <<endl;
    return 0;
}
```

I think it is obvious that there is a `goto` in the program. I had looked for a method by using `while()` to attain the goal to replay the program after illegal input for a long time. However, for a consice code, I finally used `goto`.

- Eventually, I wanted to speed up my program. First I simplified the result (quick result for existing one zero input):

```
 //forth multiply
        if(a == "0" | b == "0") result += "0";
        else result += Multiply(a,b);
```

Then, I found that `cin` and `cout` are inefficient because of buffer in the below website.

(https://stackoverflow.com/questions/1042110/using-scanf-in-c-programs-is-faster-than-using-cin)

By using `ios::sync_with_stdio(0)`, we can eliminate the buffer of `iostream`.
And by using `cin.tie(0)` and `cout.tie(0)` , we can unbind `cin` and `cout` , reduce IO burden, and further speed up thre execution efficiency.

```
ios::sync_with_stdio(0),cin.tie(0),cout.tie(0);
```

(then the code above was added into my program)

Part 5 - Summary

Although it is just an integer multiplication calculator, I still studied a lot fom it. Everytime I thought this was my ultimate edition, some ridiculous situation might occur, forcing me to optimize the code. To try best to improve the program, I had asked my friends to give some very surprising and shocking examples to test the strength of my code. I gained a lot from the goal of satisfying as many needs as possible.