به نام پروردگار قلم

فهرست مطالب:

ایده اصلی:

طبق توضیحات پروژه، باید برنامه ای نوشت که دو عدد بزرگتر از ۱۰۰ رقم را بتواند ضرب کند.

برای انجام این کار بهتر این است که از رشته ها (string) استفاده کرد چرا که محدودیتی برای دریافت کاراکتر ها ندارند. اما این را هم باید در نظر گرفت که کاربر میتواند هر کاراکتری را وارد کند پس باید دقت کنیم که در برنامه فقط از اعداد استفاده شود. همچنین برنامه باید اعداد مثبت و منفی را تشخیص دهد.

پس از دریافت اعداد باید طول آنها محاسبه شود و یک واحد از آن کم کنیم تا به هنگام دسترسی به آرایه یا رشته به خانه های حافظهٔ درست دسترسی داشته باشیم. یک آرایه میسازیم و مقدار تمام خانه های اش را برابر صفر میگذاریم و طول آن را برابر مجموع طول دو رشته قرار می دهیم. اعداد کاراکتری را به integer تبدیل می کنیم، سپس با ایجاد یک حلقه تو در تو رقم اول عدد اول در تمام ارقام عدد دوم ضرب شود و حاصل آن در خانه مجموع طول اعداد به علاوه یک آرایه ذخیره شود. در نهایت حاصل ضرب را به صورت آرایه ای داریم که حاصل ضرب به صورت رقم به رقم در خانه های آن ذخیره شده اند.

حال آرایه را به رشته تبدیل میکنیم به طوری که اگر سمت چپ عدد حاصل ضرب صفر باشد آن را در رشته ذخیره نمی کنیم و اعداد integer را به کاراکتر تبدیل می کنیم و در رشته ذخیره میکنیم.

در نهایت رشته را نمایش می دهیم.

```cpp
#include <iostream>
#include <string>
#include <vector>
using namespace std;

string bigMultiply(string , string);
int main() {
```

توضیحات بخش های برنامه:

اضافه کردن کتاب خانه ها: در بخش اول به اضافه کردن کتابخانه های مورد نیاز می پردازیم. من در این برنامه از کتابخانه های string و vector استفاده کرده ام. کتاب خانه vector شامل توابع سودمند و نوعی ساختار مشابه آرایه است. در واقع vector آرایه ای قابل انعطاف است، به طوری که می توان بعد از مشخص کردن نوع آن (integer, character, ...) می توان طول آن را تغییر داد، داده های آن را حذف کرد یا داده های جدید به آن اضافه کرد. برای حذف محدودیت های آرایه از vector استفاده کرده ام.

در ادامه تابع bigMultiply را معرفی کرده ام. این تابع دو ورودی از جنس رشته می گیرد (که همان دو عدد ما هستند) و در نهایت یک رشته (که همان حاصل ضرب است را نشان میدهد.

```cpp
string bigMultiply(string num1, string num2) {// this function multiply two large number
    if (num1 == "0" || num2 == "0")
        return "0";
    vector<int> result(num1.size() + num2.size(), 0);
    for (int i = num1.size() - 1; i >= 0; i--) {
        for (int j = num2.size() - 1; j >= 0; j--) {
            result[i + j + 1] += (num1[i] - '0') * (num2[j] - '0');
            result[i + j] += result[i + j + 1] / 10;
            result[i + j + 1] %= 10;
        }
    }
```

تابع bigMultiply : این تابع کار های زیادی را انجام می دهد که خط به خط آن را بررسی می کنیم.

در خط اول خروجی تابع را تعیین و دو پارامتر از جنس رشته با نام های num1(عدد اول) و num2 (عدد دوم) برای آن تعریف میکنیم.

در خط دوم تصویر بررسی میکنیم اگر مقدار دو رشته ۰ باشد تابع صفر را برمیگرداند.

در خط چهارم vector ای از جنس integer با نام نتیجه (result) با اندازه مجموع اندازه دو عدد(پارامتر اول) و مقدار اولیه صفر برای همه داده ها(پارامتر دوم) تعریف می کنیم.

در حلقه تو در تو در بخش اول مقدار i را برابر با یک واحد کمتر از عدد اول می گذاریم و شرط i بزرگتر مساوی صفر را برقرار می کنیم و به ازای هر بار اجرای حلقه یک واحد از i کم میکنیم. در واقع این شرط و ساختار حلقه باعث میشود که ما به کاراکتر آخر عدد اول دسترسی داشته باشیم، به ترتیب از راست به چپ کاراکتر ها را به عدد تبدیل کنیم و عمل ضرب را روی آن انجام دهیم.

در حلقه درونی j را برابر با یک واحد کمتر از طول عدد دوم قرار میدهیم و به ازای شرط j بزرگ تر مساوی صفر j را یک واحد کم میکنیم.

در خط بعد خانه i+j+1 وکتور نتیجه را با حاصل ضرب integer ای کاراکتر ها جمع میکنیم و در همان خانه ذخیره میکنیم. در اولین اجرا این خانه در واقع رقم یکان حاصل ضرب است.

سپس در خط بعد خانه i+j وکتور نتیجه با خانه خارج قسمت خانه بعدی بر ده جمع میشود.(این کار در واقع رقم نقلی را به جای درست‌اش منتقل می کند.)

در خط بعد باقی مانده خانه i+j+1 وکتور نتیجه بر ده در همان خانه قرار داده می شود.

ادامه تابع و حذف صفر های بی معنا:

در ادامه تابع ضرب یک رشته به نام strResult (رشته نتیجه)تعریف می شود و در ابتدا آن را خالی قرار می دهیم. (این کار برای این انجام میشود که در صورتی که خانه حافظه ای به این متغیر اختصاص داده شده باشد پاک شود.)

```
string strResult = "";
    bool nmZero = true;
    for (int num : result) {//for-each loop
        if (nmZero && num == 0)
        continue;
        nmZero = false;
        strResult += num + '0';
    }
    return strResult;
}
```

در ادامه متغیر nmzero (صفر بی معنا) از جنس boolean معرفی می شود و ارزش آن را درست قرار می دهیم.

در خط بعد از حلقه for مبتنی بر محدوده استفاده می شود. ساز و کار این حلقه بدین شکل است که به تعداد داده های آرایه نتیجه اجرا میشود. متغیر num از نوع integer هر بار با خانه های آرایه مقدار دهی میشود و در آغاز مقدار آن برابر با خانه اول آرایه است.

شرطی را میگذاریم در صورتی که ارزش صفر بی معنا برابر درست و مقدار num برابر صفر باشد حلقه دوباره اجرا میشود.

و اگر شرط بالا نقض شود ارزش صفر بی معنا به نادرست تغییر می یابد و به رشته نتیجه مقدار کاراکتری عدد اضافه می شود و در رشت نتیجه ذخیره می شود. در پایان تابع رشته نتیجه برگردانده می شود.(کد تابع پس از تابع اصلی نوشته شده است اما من در اینجا برای درک بهتر توضیحات‌اش را در ابتدا آورده ام.

```
string num1, num2;
cout << "Please Enter Your First Number: ";
cin >> num1;
cout << "--------------------\nPlease Enter Your Second Number: ";
cin >> num2;
int num1len=num1.size();
int num2len=num2.size();
```

بخش اول تابع اصلی: در بخش اول تابع اصلی به معرفی دو رشته num1 و num2 پرداخته ام سپس با ارائه پیام مناسب آن را از کاربر دریافت کرده ام. دو متغیر integer برای طول دو عدد معرفی و آنها را در num1len (طول عدد اول) و num2len (طول عدد دوم) ذخیره کرده ام.

```
for(int i=0;i<num1len;i++){
      if(i==0 &&num1[i]=='-')
        continue;
      if(!(num1[i]>= '0' &&num1[i]<= '9')){
        cout << "--------------------\nPlease Enter First Number Correctly!\nLike:\n1234\nYou Enter Incorrect Charcter Like:\n'a,A,~,etc'\n";
        system("pause");
        return 0;
    }
  }
  for(int i=0;i<num2len;i++){
    if(i==0 && num2[i]=='-')
        continue;
    if(!(num2[i]>='0'&&num2[i]<='9')){
        cout << "--------------------\nPlease Enter Second Number Correctly!\nLike:\n1234\nYou Enter Incorrect Charcter Like:\n'a,A,~,etc'\n";
        system("pause");
        return 0;
    }
  }
```

جلوگیری از بروز مشکلات: در این بخش با ارائه پیام مناسب از بروز مشکلاتی مانند وجود کاراکتری به جز عدد در میان کاراکتر ها جلوگیری کرده ام.

```
//negative numbers?
    bool isNegative1 = num1[0] == '-';
    bool isNegative2 = num2[0] == '-';
    if (isNegative1)
    num1 = num1.substr(1);
    if (isNegative2)
    num2 = num2.substr(1);
    string multiplyResult = bigMultiply(num1, num2);
    if (isNegative1 != isNegative2)
    multiplyResult = '-' + multiplyResult;
    cout << "Result: " << multiplyResult << endl << "--------------------\n";
  system("pause");
    return 0;
}
```

مشخص کردن اعداد منفی: در این بخش دو متغیر Boolean به نام های is negative1,2 برای دو عدد تعریف کرده ام که ارزش آنها با درستی یا نادرستی این عبارت که آیا خانه اول کاراکتر (-) هست یا نه مشخص می شود.

در خطوط بعد از تابع substr() استفاده کرده ام. این تابع در کتابخانه string موجود است و کارش این است که مقدار شروع رشته را عوض کند. در اینجا اگر عدد منفی باشد به جای خانه صفر که کاراکتر (-) قرار دارد از خانه یک که عدد است رشته شروع میشود.

در قسمت بعد رشته multiplyResult برای ذخیره نتیجه حاصل ضرب به صورت رشته تعریف شده است. در آن با فراخوانی تابع bigMultiply رشته اعداد وارد شده به تابع رفته و عملیات ضرب بر روی آنها انجام میشود.

در شرط بعدی اگر عدد اول با عدد دوم هم ارزش نباشند(یکی منفی و دیگری مثبت باشد.)کاراکتر (-) به ابتدای رشته multiplyResult افزوده میشود و در آن ذخیره میشود.

در نهایت با نمایش یک پیام مناسب نتیجه حاصل ضرب نمایش داده میشود و برنامه به پایان می رسد.

محدودیت این برنامه به اندازه محدودیت سخت افزار سیستم است!.

تصاویر چند اجرا:

```
Please Enter Your First Number: 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
--------------------
Please Enter Your Second Number: 98765432109876543210987654321098765432109876543210987654321098765432109876543210987654321098765432109876543210
Result: 12193263113702179522618503273386678859451150739156363359236761164455788599298790108215200135650052123609205801112635258986434993786160646167367779295611949397448712086533622923332237463801111263526900
--------------------
Press any key to continue . . .
```

```
Please Enter Your First Number: 12314124m123
--------------------
Please Enter Your Second Number: 123123
--------------------
Please Enter First Number Correctly!
Like:
1234
You Enter Incorrect Charcter Like:
'a,A,~,etc'
Press any key to continue . . .
```

```
Please Enter Your First Number: -1234567890
--------------------
Please Enter Your Second Number: 1234567890
Result: -15241578750190521000
--------------------
Press any key to continue . . .
```

کد نهایی:

```cpp
#include <iostream>
#include <string>
#include <vector>
using namespace std;

string bigMultiply(string , string);
int main() {
    string num1, num2;
    cout << "Please Enter Your First Number: ";
    cin >> num1;
    cout << "--------------------\nPlease Enter Your Second Number: ";
    cin >> num2;
    int num1len=num1.size();
    int num2len=num2.size();
    for(int i=0;i<num1len;i++){
      if(i==0 &&num1[i]=='-')
        continue;
      if(!(num1[i]>= '0' &&num1[i]<= '9')){
        cout << "--------------------\nPlease Enter First Number Correctly!\nLike:\n1234\nYou Enter Incorrect Charcter Like:\n'a,A,~,etc'\n";
        system("pause");
        return 0;
    }
  }
  for(int i=0;i<num2len;i++){
    if(i==0 && num2[i]=='-')
        continue;
    if(!(num2[i]>='0'&&num2[i]<='9')){
        cout << "--------------------\nPlease Enter Second Number Correctly!\nLike:\n1234\nYou Enter Incorrect Charcter Like:\n'a,A,~,etc'\n";
        system("pause");
        return 0;
    }
  }
    //negative numbers?
    bool isNegative1 = num1[0] == '-';
    bool isNegative2 = num2[0] == '-';
    if (isNegative1)
    num1 = num1.substr(1);
    if (isNegative2)
    num2 = num2.substr(1);
    string multiplyResult = bigMultiply(num1, num2);
    if (isNegative1 != isNegative2)
    multiplyResult = '-' + multiplyResult;
    cout << "Result: " << multiplyResult << endl << "--------------------\n";
  system("pause");
    return 0;
}
string bigMultiply(string num1, string num2) {// this function multiply two large number
    if (num1 == "0" || num2 == "0")
    return "0";
    vector<int> result(num1.size() + num2.size(), 0);
    for (int i = num1.size() - 1; i >= 0; i--) {
        for (int j = num2.size() - 1; j >= 0; j--) {
            result[i + j + 1] += (num1[i] - '0') * (num2[j] - '0');
            result[i + j] += result[i + j + 1] / 10;
            result[i + j + 1] %= 10;
        }
    }
    string strResult = "";
    bool nmZero = true;
    for (int num : result) {//for-each loop
        if (nmZero && num == 0)
    continue;
        nmZero = false;
        strResult += num + '0';
    }
    return strResult;
}
```