**Big Number Addition:**

#include <iostream>

#include <vector>

#include <map>

#include <cstring>

#include <list>

#include <queue>

#include <cmath>

#include <cstdio>

#include <algorithm>

#include <stack>

#include <sstream>

#include <bitset>

#include <set>

#define sf scanf

#define pf printf

using namespace std;

int main()

{

vector<int> a, b,c;

char value1, value2;

while(sf("%c",&value1)){

if(value1== '\n'){

break;

}

a.push\_back(value1-'0');

}

while(sf("%c",&value2)){

if(value2=='\n'){

break;

}

b.push\_back(value2-'0');

}

vector<int>::reverse\_iterator rit1,rit2 ;

int carry=0, v1, v2, c1=0, c2=0;

rit2=b.rbegin();

c1 =a.size()>b.size() ? a.size() : b.size();

for(rit1=a.rbegin(); c2<c1; rit1++)

{

if(c2>=a.size())

v1=0;

else

v1=\*rit1;

if(c2>=b.size())

v2=0;

else

v2=\*rit2;

if((v1+v2+carry)>=10){

c.push\_back((v1+v2+carry)%10);

carry=1;

}

else{

c.push\_back(v1+v2+carry);

carry=0;

}

rit2++;

c2++;

}

if(carry>0)

c.push\_back(1);

for(rit1=c.rbegin(); rit1!=c.rend(); rit1++)

pf("%d",\*rit1);

return 0;

}

**Big Number Substruct:**

#include <iostream>

#include <vector>

#include <map>

#include <cstring>

#include <list>

#include <queue>

#include <cmath>

#include <cstdio>

#include <algorithm>

#include <stack>

#include <sstream>

#include <bitset>

#include <set>

#define sf scanf

#define pf printf

using namespace std;

int main()

{

vector<int> a, b,c;

char value1, value2;

while(sf("%c",&value1)){

if(value1== '\n'){

break;

}

a.push\_back(value1-'0');

}

while(sf("%c",&value2)){

if(value2=='\n'){

break;

}

b.push\_back(value2-'0');

}

vector<int>::reverse\_iterator rit1,rit2 ;

int carry=0, v1, v2, c1=0, c2=0;

rit2=b.rbegin();

c1 =a.size()>b.size() ? a.size() : b.size();

for(rit1=a.rbegin(); c2<c1; rit1++)

{

if(c2>=a.size())

v1=0;

else

v1=\*rit1;

if(c2>=b.size())

v2=0;

else

v2=\*rit2;

// pf("\nv1: %d, v2: %d, Carry: %d\n",v1,v2,carry);

if(v1>=(v2+carry)){

c.push\_back(v1-(carry+v2));

carry=0;

}

else{

c.push\_back(((v1+10)-(carry+v2)));

carry=1;

}

rit2++;

c2++;

}

//For erasing leading zeros....

/\*\*{

int count=0;

vector<int>::reverse\_iterator cit;

for(cit=c.rbegin(); ;cit++)

{

if(\*cit == 0)

count++;

else

break;

}

c.erase(c.end()-count ,c.end());

}\*\*/

for(rit1=c.rbegin(); rit1!=c.rend(); rit1++)

pf("%d",\*rit1);

return 0;

}

**Big Number Multiplication:**

#include <iostream>

#include <fstream>

#include <vector>

#include <map>

#include <cstring>

#include <list>

#include <queue>

#include <cmath>

#include <cstdio>

#include <algorithm>

#include <stack>

#include <sstream>

#include <bitset>

#include <set>

#define sf scanf

#define pf printf

using namespace std;

vector<int> reverse\_vector(vector<int> l)

{

vector<int> temp;

vector<int>::reverse\_iterator r;

for(r=l.rbegin();r!=l.rend();r++)

temp.push\_back(\*r);

return temp;

}

vector<int> add\_vector(vector<int> p, vector<int>q)

{

vector<int>c;

vector<int>::reverse\_iterator ritp,ritq ;

int carry=0, v1, v2, c1=0, c2=0;

ritq=q.rbegin();

c1 =p.size()>q.size() ? p.size() : q.size();

for(ritp=p.rbegin(); c2<c1; ritp++)

{

if(c2>=p.size())

v1=0;

else

v1=\*ritp;

if(c2>=q.size())

v2=0;

else

v2=\*ritq;

if((v1+v2+carry)>=10){

c.push\_back((v1+v2+carry)%10);

carry=1;

}

else{

c.push\_back(v1+v2+carry);

carry=0;

}

ritq++;

c2++;

}

if(carry>0)

c.push\_back(1);

return c;

}

int main()

{

vector<int>a,b;

char value1,value2;

while(sf("%c",&value1)){

if(value1== '\n'){

break;

}

a.push\_back(value1-'0');

}

while(sf("%c",&value2)){

if(value2=='\n'){

break;

}

b.push\_back(value2-'0');

}

vector<int>::reverse\_iterator rit1,rit2 ;

vector<int>f ;

f.push\_back(0);

int count=0;

for(rit2=b.rbegin(); rit2!=b.rend(); rit2++)

{

vector<int>c;

int carry=0;

for(rit1=a.rbegin(); rit1!=a.rend(); rit1++)

{

int p,q, r;

p =\*rit1;

q=\*rit2;

c.push\_back( ((p\*q)+carry)%10);

carry = ((p\*q)+carry)/10;

}

if(carry>0)

c.push\_back(carry);

c=reverse\_vector(c);

if(count>0){

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

c.push\_back(0);

}

c=add\_vector(f,c);

f=reverse\_vector(c);

count++;

}

vector<int>::iterator it;

for(int i=1;i<=f.size()+3;i++)

cout<<"-";

cout<<endl;

cout<<"[\*]";

for(it=f.begin(); it!=f.end();it++)

cout<<\*it;

return 0;

}

**Big Number Dividation:**

package descreateMath;

import java.math.BigInteger;

import java.util.Scanner;

public class BigIntegerDivision {

public static void main(String args[]){

System.out.println("Enter your Input1: ");

Scanner input = new Scanner(System.in);

BigInteger number = input.nextBigInteger();

System.out.println("Enter your Input2: ");

BigInteger number2 = input.nextBigInteger();

BigInteger result, reminder;

result = number.divide(number2);

reminder = number.remainder(number2);

System.out.println(""+number+" / "+number2+" = "+result);

System.out.println(""+number+" % "+number2+" = "+reminder);

}

}