**Matrix Expo, O(d^3 lg(n))**

****

mat base res

struct matrix{

int row,col,cell[2][2]; //maximum cell size, increase row-col for constant

};

matrix res, base, mat, ID, A, temp;

matrix mult(matrix a,matrix b)

{

temp.row=a.row;

temp.col=b.col;

for(int i=0;i<a.row;i++)

for(int j=0;j<b.col;j++)

{

int sum=0;

for(int k=0;k<a.col;k++)

sum=(sum+(((a.cell[i][k]%MOD)\*(b.cell[k][j]%MOD))%MOD))%MOD;

temp.cell[i][j]=sum;

}

return temp;

}

matrix mat\_expo(matrix a,int n)

{

matrix ret=ID;

while(n)

{

if(n & 1) ret=mult(ret,a);

n>>=1;

a=mult(a,a);

}

return ret;

}

int main()

{

int tc,kk=1,a,b,n,m;

for(int i=0;i<2;i++)

for(int j=0;j<2;j++)

{

mat.cell[i][j]=base.cell[i][j]=0;

if(i==j) ID.cell[i][j]=1;

else ID.cell[i][j]=0;

}

//change row-col here for ID, base, mat, res

mat.row=base.row=base.col=ID.row=ID.col=res.row=2;

res.col=mat.col=1;

//change base here.

base.cell[0][1]=base.cell[1][0]=base.cell[1][1]=1;

cin>>tc;

while(tc--)

{

cin>>a>>b>>n;

//change mat here. Change base if needed.

mat.cell[0][0]=a%MOD;

mat.cell[1][0]=b%MOD;

if(n==0)

cout<<a%MOD<<endl;

else if(n==1)

cout<<b%MOD<<endl;

else

{

A=mat\_expo(base, n-1);

res=mult(A, mat);

cout<<res.cell[1][0]%MOD<<endl;

}

}

return 0;

}