**01串：**

#include <iostream>

using namespace std;

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

#include <algorithm>

bool check(int \*a,int num,int len)

{

int cnt=0;

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

{

if(a[i]==0&&a[i+1]==1)

{

cnt++;

}

}

if(cnt>=num)

return true;

else return false;

}

int main()

{

int n,m;

while(scanf("%d%d",&n,&m)!=EOF)

{

int Tol=0;

for(int i=m;i<=n-m;i++)

{

int a[100]={0};

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

{

a[n-j]=1;

}

do{

if(check(a,m,n))

{

Tol++;

}

}while(next\_permutation(a,a+n));

}

printf("%d\n",Tol);

}

return 0;

}

**第四题：**

**斐波拉矩阵**

#include <iostream>

#include <stdio.h>

#include <string.h>

using namespace std;

struct Matrix{

int d[2][2];

}ans,ori;

int n,m;

Matrix mutil(Matrix &lhs,Matrix &rhs)

{

Matrix ret;

memset(ret.d,0,sizeof(ret.d));

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

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

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

{

ret.d[i][j]+=lhs.d[i][k]\*rhs.d[k][j];

if(ret.d[i][j]>=m)

ret.d[i][j]%=m;

}

return ret;

}

void solve()

{

int k=n;

while(k){

if(k&1)

ans=mutil(ans,ori);

k>>=1;

ori=mutil(ori,ori);

}

}

int main()

{

int t;

scanf("%d",&t);

while(t--)

{

scanf("%d%d",&n,&m);

ans.d[0][0]=ans.d[1][1]=1;

ans.d[0][1]=ans.d[1][0]=0;

ori.d[0][0]=2;

ori.d[0][1]=ori.d[1][0]=ori.d[1][1]=1;

solve();

printf("%d\n",ans.d[0][1]);

}

//

cout << "Hello world!" << endl;

return 0;

}