String01 (loại bỏ xâu con)

```
#include<bits/stdc++.h>
using namespace std;
int main(){
   string s,sub;  // khởi tạo 2 biến string s,sub;
   getline(cin,s); // hàm nhập chuỗi mẹ
   getline(cin,sub);// hàm nhập chuỗi con
        int pos=s.find(sub);// trả về vị trí đầu tiên xuất hiện của chuỗi con trong chuỗi mẹ
        s.erase(pos,sub.size());//xoa đi chuỗi con trong chuỗi mẹ
        cout<<s<<endl;// xuất ra chuỗi mẹ sau khi xóa chuỗi con
}
```

String 02(Địa chỉ email ptit)

```
#include<iostream>
#include<string>
using namespace std;
int main() {
        string s=""; string s1;//khởi tạo 2 chuỗi s,s1
        getline(cin, s1);// nhập chuỗi s1;
        for (long long i = s1.size() - 1; i >= 0; i--) {
                if (s1[i] >= 65 && s1[i] <= 90)  s1[i] += 32;//biến các ký tự in hoa sang in thường
        }
```

```
    long long k = 0;int kt=0;
    for (long long i = s1.size() - 1; i >= 0; i--) {// duyệt chuỗi từ cuối về đầu,tìm vị trí khoảng trắng đầu tiên để chèn @vào
        if (s1[i] != ' ' && s1[i - 1] == ' ') {
            k = i;
            break;
        }
        else if(i==0) kt=1;k=i;// nếu không tồn tại khoảng trắng nghĩa là tên chỉ có 1 từ(I sẽ chạy về 0)
    }
    for (long long i = k; i < s1.size(); i++)     s += s1[i];// lấy được tên
    if(kt==0) s += s1[0];// lấy được chữ cái đầu tiên của tên họ
    for (long long i = 0; i < k-1; i++) {
        if (s1[i] == ' ' && s1[i + 1] != ' ') s += s1[i + 1];// ta sẽ lấy lần lượt đc chữ cái đầ tiên của mỗi từ
    }
    s += "@ptit.edu.vn";// chèm them "@ptit.edu.vn vào cuối chuỗi s
    cout << s<<endl;// xuất chuỗi s
    return 0;
}
```

String 03 Chuẩn hóa tên

// Các bước làm bài này của mình là như sau

1, nếu tồn tại ký tự khoảng trắng ở đầu chuỗi → xóa tất cả các ký tự khoảng trắng đó

2,xóa tất cả các ký tự khoảng trắng tồn tại ở khoảng giữa 2 từ của tên, chỉ giữ lại 1 ký tự khoảng trắng

3,chuyển tất cả các ký tự in hoa sang in thường(dùng vòng lặp duyệt từ đầu đến cuối nếu có ký tự là in hoa ta sẽ chuyển sang in thương)

4, tim vị trí để điền dấu , duyệt vong lặp for từ đầu xuống, tìm vị trí khoảng trắng đầu tiên đè chèn dấu ,(nếu tên chỉ có 1 từ thì I sẽ chạy đến -1, lúc đó t chỉ cần chen ',' vào vị trí s[0]

→Nếu b chưa làm được thì theo hương dẫn trên bạn hoàn toàn có thể tự code lại theo cách riêng cua mình mà ko cần đọc code của mình vì sẽ khó hiểu

```
#include<iostream>
#include<cstring>
using namespace std;
int main(){
	string s;getline(cin,s);
	while(s[0]==' ') s.erase(s.begin());
	for(int i=0;i<s.size();i++){
		if(s[i]==' '&&s[i+1]==' '){
			s.erase(s.begin()+i+1);
			i--;
		}
	}
	int mark=0;
	for(int i=0;i<s.size();i++){
		if(s[i]==' '&&((s[i+1]>=65&&s[i+1]<=90)||(s[i+1]>=97&&s[i+1]<=122))) mark=1;
	}
```

```
for(int i=0;i<s.size();i++){
if(s[i]>=65&&s[i]<=90) s[i]+=32;
}
int j=0;int kt=1;

for(int i=s.size()-1;i>=0;i--){
        if(s[i]!=' '&&s[i-1]==' '){
                j=i;
                s.insert(s.begin()+j-1,',');
                kt=0;
                break;
        }
}
if(j==0) s.insert(s.begin(),',');
if(s[0]!=',') s[0]-=32;
for(int i=j+1;i<s.size();i++)          s[i]-=32;
for(int i=1;i<j;i++){
        if(s[i]==' ') s[i+1]-=32;
}
cout<<s;


}
```

String 1

```
#include<bits/stdc++.h>
using namespace std;
string s;int k;
void input(){
   cin >> s >> k;// k chính là số k tự thêm vào để trơ thành dãy pangram
}
void solve(){
   int h[26] = {0} , t[26] = {0};  // khai báo 2 mảng mục đích để kiêm tra xem những ký tư nào đã xuất hiện trong xâu, và từ đó tim ra số ký tự còn thiếu để tham vào thanh dãy Pangram
   for(int i = 0 ; i < s.length() ; i++){
      if ( 'a' <= s[i] && s[i] <= 'z' )
         h[s[i] - 'a']++;
      else t[s[i] - 'A']++;
   }
   int t1 = 0 , h1 = 0 ;
   for(int i = 0 ; i < 26 ; i++){
      if (t[i] > 0) t1++;// t chính là số lượng ký tự thường trong bản chữ cái (mỗi kỹ tự đếm 1 lần) xuất hiện trong dãy
      if (h[i] > 0) h1++;++;// t chính là số lượng ký tự hoa trong bản chữ cái (mỗi kỹ tự đếm 1 lần) xuất hiện trong dãy
   }
   if( max(t1,h1) + k >= 26) cout << 1 << endl;// ta sẽ tìm đc số nhỏ nhất ký tự cần them vào để chuỗi trờ thanh chuỗi Pangram → so sánh với k để đưa kết luận
   else cout << 0 << endl;
}
int main(){
```

```
    int t=1;
    cin>>t; cin.ignore();// chú ý dòng này
    while(t--){
        input();solve();
    }
    return 0;
}
```

String 2

//dấu hiệu chia hết cho 11 là giá trị tuyệt đối của hiêu giữa 2 tổng các chữ số ở vị trí chăn và vị trí lẻ chia hết cho 11 thì số đó chia hết cho 11

Lưu ý các chuyển ký tự số '6' sang số tự nhiên là 6 ta chỉ cần lấy ki tự trừ đi ký tự 0 (-'0')

```
#include<iostream>
#include<cmath>
using namespace std;
int main(){
        int t;cin>>t;
        while(t--){
        string s;cin>>s;
        int sum1=0,sum2=0;
        for(int i=0;i<s.size();i++){
                if(i%2==0){
                        sum1=sum1+(s[i]-'0');// tổng các chữ số ở vị trí chăn
                }
                else sum2=sum2+(s[i]-'0');// tổng các chữ số ở vị trí lẻ
```

```
        }
        int x=abs(sum1-sum2);// giá trị tuyệt đối của 2 tổng
        if(x%11==0) cout<<1;
        else cout<<0;
        cout<<endl;
        }
return 0;
}
```

String 3

// bài này sẽ có 1 số cách làm khác ngăn hơn nhưng mình sẽ làm theo các đúng như suy nghĩ nhiều người

1, chuyển số nhị phân về số thập phân, nếu chứ số tận cũng là 0 hoặc 5 thì số đó chia hết cho 5(ý tưởng là vậy, nhưng để làm đúng thì phải cần thận)

2, vì giá trị n đề cho là rất lớn nên ta dùng kiểu string

3, sau khi chuyển về số thập phân ta chỉ cần quan tâm số tận cùng(nên các bước giai đoạn ta nên chia mod 10 để sum trả về số tận cùng)

```
#include<iostream>
#include<cmath>
using namespace std;
long long luyThua(int a,int n){
        if(n==0) return 1;
        long long m=luyThua(a,n/2);
        m*=m;
```

```
        m%=10;
        if(n%2==0) return m;
        return (m*a)%10;
//hàm mũ này giúp các bạn tìm đc lũy thừa trong 1 số bài toán chia mod, chống tràn
}
//bên trên là 1 hàm lũy thừa(có chia mod 10 giúp chung ta tránh bị tràn số)
int main(){
        long long t;cin>>t;
        while(t--){
        string s;cin>>s;
        long long dem=0;long long sum=0;
        for(long long i=s.size()-1;i>=0;i--){
                long long m=(s[i]-'0')*luyThua(2,dem);
                sum=sum+m%10;
                while(sum>=10){   // vì ta chỉ cần chữ số cuối cùng thôi
                        sum%=10;
                }

                dem++;
        }
        //cout<<sum<<endl;
        if(sum==0||sum==5) cout<<"Yes";// nếu số cuối cùng =0 hoặc 5 thì in yes
        else cout<<"No";
        cout<<endl;
```

```
	}
return 0;
}
```

String 14:

```
#include<iostream>
using namespace std;
int main(){
	int t;cin>>t;
	while(t--){
		cin.ignore();
		string s;
		getline(cin,s);
		int count=1;
		for(int i=0;i<s.size();i++){
			if(s[i]==' '||s[i]=='\t'||s[i]=='\n'){
				count++;// đếm từ
			}
		}
		cout<<count<<endl;
	}
return 0;
}
```

String 15

```
// bài này đơn giản, chỉ cần 2 vong lặp for để đêm
```

```
#include<iostream>
#include<string>
#include<vector>
#include<algorithm>
using namespace std;
int main(){
	int t;cin>>t;
	while(t--){
		string s;cin>>s;
		int count=s.size();
		for(int i=0;i<s.size();i++){
			for(int j=i+1;j<s.size();j++){
				if(s[j]==s[i]){ // chuỗi con có ký tự đầu và ký tự cuối giống nhau
					count++;
				}
			}
		}
		cout<<count<<endl;
	}

return 0;
}
```

String 17;

```
// bài này khá đơn giản
#include<iostream>
#include<cstring>
using namespace std;
int main(){
	int t;cin>>t;
	cin.ignore();
	while(t--){
		string s;getline(cin,s);
		int dem[100000]={0};
		for(int i=0;i<s.size();i++) dem[s[i]]++;
		for(int i=0;i<s.size();i++){
			if(dem[s[i]]==1) cout<<s[i];
		}
		cout<<endl;
	}
	return 0;
}
```

String 20

```
#include<iostream>
using namespace std;
int main() {
   long long t; cin >> t;
   cin.ignore();// chú ý dòng này, nếu ko có bạn sẽ ko nhập đc bộ test tiếp theo
```

```
    while (t--) {
        long long start;
        string s;// chuỗi ban đầu
        getline(cin, s);
        long long i = s.size() - 1;
        string res = "";// khởi tạo chuôĩ để xử lý
        long long end = s.size();// khỏi tạo giá trị kết thúc của vong lặp
        while (i >= 0) {
            if (s[i] == ' ') {// tìm ký tự khoảng trắng đầu tiên
                start = i + 1;// sau khi xác định vị trí của khoảng trắng, ta sẽ lấy đc 1 từ của chuỗi mẹ từ vị trí i+1 đến vị trí end
                while (start != end) {// thực hiện vong lặp để lấy từng từ của chuỗi mẹ theo chiều từ phải qua trái
                    res += s[start++];// cộng từng kỳ tự của s từ khoảng trắng cho đến hết 1 từ vào chuỗi res;
                }
                res += " ";// cộng khoảng trắng sau khi lấy được 1 từ
                end = i;// đặt biến kết thuc vong lặp bằng vị trí khoảng trắng trước đó

            }
            i--;
        }
// sau khi thực hiện đoạn code trên ta vẫn còn chưa in ra đc hết, vẫn còn thiếu 1 từ
        start = 0;
        while (start != end) {// thực hiện vong lặp này sẽ in nốt từ còn lại
```

```
        res += s[start++];
      }
      cout << res << endl;
   }
   return 0;
}
```

String 5

```
#include <iostream>
using namespace std;
        int main(){
                int t;cin>>t;
                while(t--){
                        string s;cin>>s;
                        int d[300]={0};int res=0;
                        for(int i=0;i<s.size();i++){
                                d[s[i]]++;
                                if(res<d[s[i]]){
                                        res=d[s[i]];
                                 }
                         }
                         if(res<=s.size()-res&&s.size()%2==0)cout <<1;
                         else if((res<(s.size()/2+1))&&s.size()%2==1) cout<<1;
                         else cout<<0;
                         cout<<endl;
```

```
        }

        return 0;
}
```

String 6

```
//để thu được tổng lớn nhấtt thì ta sẽ đổi tất cả ký tự '5' → '6' sau đó tính tổng
//để thu được tổng nhỏ nhấtt thì ta sẽ đổi tất cả ký tự '6 → '5 sau đó tính tổng
#include<bits/stdc++.h>
using namespace std;
int main(){
    int t;cin>>t;
    while(t--) {
        string s1,s2;
    cin>>s1>>s2;
    int f[26]={};
    for(int i=0,j=0;i<s1.size(),j<s2.size();i++,j++){
        if(s1[i]=='6' ) s1[i]='5';// chuyển tất cả các ký tự cúa s1 là 6 thanh 5
        if(s2[j]=='6' ) s2[j]='5';// chuyển tất cả các ký tự của s2 là 6 thành 5
    }
    long long sum1=0,sum2=0;
    for(int i=0,j=0;i<s1.size(),j<s2.size();i++,j++){
        sum1=sum1*10+s1[i]-'0'; //chuyển tất cả các ký tự cúa s1 là 6 -> 5

        sum2=sum2*10+s2[j]-'0'; //chuyển tất cả các ký tự cúa s1 là 6 -> 5
```

```
        }
        cout<<sum1+sum2<<' ';
// tương tự ta có thể tim đc max của 2 tổng
        for(int i=0,j=0;i<s1.size(),j<s2.size();i++,j++){
                if(s1[i]=='5' ) s1[i]='6';
                if(s2[j]=='5' ) s2[j]='6';
        }
        sum1=0,sum2=0;
        for(int i=0,j=0;i<s1.size(),j<s2.size();i++,j++){
                sum1=sum1*10+s1[i]-'0';
                sum2=sum2*10+s2[j]-'0';
        }
        cout<<sum1+sum2<<' '<<endl;
        }
        return 0;
}
```

String 8

```
#include <bits/stdc++.h>
using namespace std;
int main(){
        int t;cin>>t;
        while(t--){
                string s;cin>>s;long long sum=0;long long res=0;
```

```
            for(int i=0;i<s.size()+1;i++){        // lưu ý, i<;i<s.size()+1 mục đích để cộng nột số cuối cung(nếu số đó là số kết thúc xâu) để cho vong lặp else bên dưới xảy ra
                if(s[i]>='0'&&s[i]<='9'){
                    sum=sum*10+(s[i]-'0');
                }
                else {
                res+=sum;sum=0;
                }
            }
            cout<<res<<endl;
        }
    return 0;
}
```

String 9

```
// bài string 9 khá giống string 8
#include<iostream>
using namespace std;
int main()
{
    int t; cin >> t;
    while (t--) {
        string s;
        cin>>s;
        int sum=0,res=0;
```

```
    for(int i=0;i<s.size()+1;i++){
      if(s[i]>='0'&&s[i]<='9'){
            sum=sum*10+s[i]-'0';
                  }
                  else{
                        res=(sum>res) ?sum:res;// điều kiện 3 ngôi
                        sum=0;
                  }
            }
            cout<<res<<endl;

  }
  return 0;
}
```

String 4

// ý tưởng của bài nay :

Để tim số xâu con chia hết cho 8 mà không chia hết cho 3, ta sẽ tìm số xâu con chia hết cho 8 trừ đi số xâu con chia hết cho 3(đó là mấu chốt giải quyết bài toán này)

Viêc đếm xâu con chia hết cho 1 số bất kỳ thì ta chỉ cần dung 2 vong lặp for như bên dưới

```
#include <iostream>
using namespace std;
string s;
 int GiaiQuyet(int k)
```

```
{
int dem=0;
for(int i=0;i<s.length();i++){
 int n=0;
        for(int j=i;j<s.length();j++){
                n=n*10+s[j]-'0'; n=n%k;
               if(n==0) dem=dem+1;
         }
}
 return dem;
}
 int main(){
  int T;cin>>T;
  while(T--){
   cin>>s;
        cout<<GiaiQuyet(8)-GiaiQuyet(24)<<endl;
        }
        return 0;
        }
```

String 21

```
#include<iostream>
#include<algorithm>
```

// việc sắp xếp các ký tự trong s theo quy tắc bảng chữ cái thì khá đơn gian vì chỉ cần dùng 1 hàm sort(s1.begin(),s1.end());

Còn việc tính tổng các chữ số có mặt trong xâu thì có hăn 2 bài string 8 string 9 để tính nên mình cũng ko bình luận gì thêm nữa

```
using namespace std;
int main(){
	cin.tie(0); ios_base::sync_with_stdio(false);
	int t;cin>>t;
	while(t--){
		int dem=0,sum=0,j=0;
		string s1="",s2="";
		cin>>s1;
		for(int i=0;i<s1.size();i++){
			if(s1[i]>='0'&&s1[i]<='9'){
				sum=sum+(s1[i]-'0');
				s1.erase(s1.begin()+i);
				i--;
			}
		}
		sort(s1.begin(),s1.end());
		cout<<s1;
		cout<<sum<<endl;
	}
	return 0;
}
```

String 13

```
#include<bits/stdc++.h>
using namespace std;
int main(){
	int t;cin>>t;
	while(t--){
		string s;cin>>s;
		int k;cin>>k;int cnt[1000];//cnt dùng làm mảng đánh dấu
		int count=0;int res=0;
		for(int i=0;i<s.size();i++){//để đếm các xâu con liên tiếp ta sẽ dùng 2 vong lặp for (i=0,j=i)
			memset(cnt,0,sizeof(cnt));// khởi tạo tất cả các phần tư của mảng cnt bằng 0
			count=0;// biến này dùng để đếm các phần từ khác nhau trong xâu con
			for(int j=i;j<s.size();j++){
				if(cnt[s[j]-'a']==0){// có nghĩa là ký tự s[j] chưa đc lặp lại lần nào hay đó là ký tự khác nhau đầu tiên
					count++; // tăng biến count lên dể đếm ký tự đó
				}
				cnt[s[j]-'a']++;// đánh dấu ký tự s[j] là đã lặp lại trước đó rồi
				if(count==k) res++;// nếu count =k có nghĩa là đã tìm được 1 xâu con có k ký tự khác nhau, tăng biến đếm res
				else if(count>k){// xâuu con  đó không còn thỏa mãn có k ký tự khác nhau
					break;// thoat vong lặp for j
```

```
                    }
                }
            }
            cout<<res<<endl;// in ra kết quả

        }
}
```

String 18

```
#include<bits/stdc++.h>
using namespace std;
int ans = 0;
string s;
void stringRemove(){
    for(int i=0; i<s.length(); i++){
        if(s[i] == '1' && s[i+1] == '0' && s[i+2] == '0'){//tim xâu con"100" trong xâu mẹ
            ans += 3;// đếm số lượng ký tự bị xâu con bị loại bỏ
            s.erase(i, 3);// xoa đi xâu con "100" từ xâu mẹ( xóa tự vị trí thứ i và xóa đi 3 ký tự)
            stringRemove();// đệ quy để tìm và xóa tiếp các xâu con "100"
        }
    }
}
int main(){
    cin.tie(0); ios_base::sync_with_stdio(false);
```

```
    int test; cin >> test;
    while(test--) {
        ans = 0;//khởi tạo biến đếm =0 sau mỗi bộ test
        cin >> s;
        stringRemove();
        cout << ans << endl;
    }
    return 0;
}
```

String 11

```
#include<bits/stdc++.h>
using namespace std;
//hàm bool để kiểm tra xem các ký tự của xâu s2 có mặt trong xâu s1 hay không, vdu trong xau s2 có 2 ký tự 'a' thì xong xâu s1 số lượng ký tự 'a' cũng phải lơn hơn hoặc bằng 2
bool subStr2ofstr1(string str1, string substr) {
    map<char, int>mpstr1, mpsubstr;// khai báo 2 map mpstr1, mpsubstr
    for (int i = 0; i < str1.length(); i++) mpstr1[str1[i]]++;// đêm số lần xuất hiện của từng ký tự trong str1
    for (int i = 0; i < substr.length(); i++) mpsubstr[substr[i]]++;// đếm số lần xuất hiện của từng ký tự trong substr
                                    //map[key]=value
    for (map<char, int>::iterator it= mpsubstr.begin(); it != mpsubstr.end(); it++)
{
        if (mpstr1[it->first] < mpsubstr[it->first]) return false;
```

```
        }
        return true;

}
int main() {
        cin.tie(0); ios_base::sync_with_stdio(false);
        int t; cin >> t;
        while (t--) {
                string str1, subStr;
                cin >> str1 >> subStr;
                int MIN = 150;// khai báo MIN >100là đc, ta chọn MIN=150
                string ans;// chuỗi kết quả
                bool check = false;
                for (int i = 0; i < str1.length(); i++) {
                        string res;
                        for (int j = i; j < str1.length(); j++) {
                                res.push_back(str1[j]);
                                if (subStr2ofstr1(res, subStr) == true && res.length() <
MIN) {
                                        MIN = res.size();//tìm chuỗi ans có độ dài ngắn
nhất
                                        ans = res;// cập nhập lại chuỗi ans
                                        check = true;// đánh dấu là tồn tại kết quả bài
toán
                                }
```

```
                }
            }
            if (check == true) cout << ans;
            else cout << -1;// nếu không tồn tại kết quả thì in -1
            cout << endl;
        }
        return 0;
}
```

String 12

// bài 12 này mình dùng ý tưởng giống bài 11,

Bước 1 ta tìm 1 xâu S1 chứa các ký tự xuất hiện trong xâu mẹ S :vidu xâu mẹ S=aadcc thì xâu S1 là adc

Bước 2 đưa bài toán về bài string 11 tìm xâu con ngắn nhất của S chứa đầy đủ các ký tự của s1

```
#include<bits/stdc++.h>
using namespace std;
const int oo = 300;
bool kiemtra(string s1,string res){
        map<char,int> maps1,mapres;
        for(int i=0;i<s1.size();i++) maps1[s1[i]]++;
        for(int i=0;i<res.size();i++) mapres[res[i]]++;
        for(map<char,int>::iterator it=mapres.begin();it!=mapres.end();it++){
            if(maps1[it->first]<mapres[it->first]) return false;
        }
        return true;
}
int main(){
```

```
    cin.tie(0); ios_base::sync_with_stdio(false);
    int t;cin>>t;
    while(t--){
        string s;cin>>s;
        set<char> st;
    for(int i=0; i<s.length(); i++) st.insert(s[i]);
    string s1(st.begin(), st.end());
        int MIN=s.size();
        for(int i=0;i<s.size();i++){
            string res="";
            int kt=0;
            for(int j=i;j<s.size();j++){
                res.push_back(s[j]);
                if(kiemtra(res,s1)&&res.size()<MIN){
                    MIN=res.size();
                }

            }
        }
        cout<<MIN<<endl;
    }
}
```

// một cách làm khác nữa của B **Cấn Ngọc Binh**

```
#include<iostream>
#include<algorithm>
#include<cmath>
#include<vector>
```

```
#include<string>
#include<map>
#include<set>
#include<sstream>
#define ll long long
using namespace std;
const int oo = 300;
// check xem trong string s co ki tu c ko
bool isHave(string s, char c){
	for(int i=0; i<s.length(); i++) if(s[i] == c) return true;
	return false;
}
// check xem xau con da du ki tu cua xau s chua
bool fullString(int A[], string s1){
	for(int i=0; i<s1.length(); i++) if(A[(int)s1.at(i)] == -1) return false;
	return true;
}
int main(){
   cin.tie(0); ios_base::sync_with_stdio(false);
   int test; cin >> test;
   while(test--){
	bool check = false;
	int A[oo] = {0}; // mang danh dau cac ki tu
	for(int i=0; i<oo; i++) A[i] = -1;
	string s; cin >> s;
	set<char> st;
	for(int i=0; i<s.length(); i++) st.insert(s[i]);
```

```
        string s1(st.begin(), st.end());
        int start = 0, end = 0, ans = 1e5;
        for(int i=0; i<s.length(); i++){
            if(isHave(s1, s.at(i))){
                A[(int) s.at(i)] = i; // cap nhat lai vi tri cua cua ki tu s[i];
                if(fullString(A, s1)){
                    int local = 1e9;
                    for(int j=0; j<s1.length(); j++){
                        local = min(local, A[(int)s1.at(j)]); // khoang cach
                        }
                        ans = min(ans, i - local + 1);
                    }
                }
            }
            cout << ans << endl;
    }
}
```

String 19 <Cấn Ngọc Binh>

```
#include<iostream>
#include<string>
#include<string.h>
#define ll long long
using namespace std;
ll F[105][105] = {0};
// F[i][j] so chuoi con xet den vi tri thu i chia du cho mod duoc j
```

```
// so 34 % 5 = 4 khi them so 6 vao cuoi ta dc 346 % 5 = 1 (hay ta lay so du = 4*10 + 6 = 46 % 5 = 1)
int main(){
	int test; cin >> test;
	while(test--){
		ll mod, k; string s;
		cin >> k >> mod >> s;
		memset(F, 0, sizeof(F));
		for(int i=1; i<=s.length(); i++){
			F[i][(s[i-1] - '0') % mod] = 1;
			for(ll j=0; j<mod; j++){
				F[i][j] += F[i-1][j]; // neu ko chon ki tu thu i
				F[i][(j*10 + s[i-1] - '0') % mod] += F[i-1][j]; // neu chon ki tu thi i
			}
		}
		cout << F[k][0] << endl;
	}
}
```

String 16<Cấn Ngọc Binh>

```
#include<iostream>
#include<string>
#include<string.h>
#define ll long long
using namespace std;
```

```
ll n, r, b, g, A[100], len, sum = 0;
ll fact, ans = 0;
// tinh giai thua k!
/*
test: 4 1 1 1
1 1 2 = 4! / 2! = 12
1 2 1 = 4! / 2! = 12
2 1 1 = 4! / 2! = 12
-> ans = 36
*/
ll factorial(int k){
	ll res = 1;
	for(int i=1; i<=k; i++) res *= i;
	return res;
}
void out(int p){
	ll tmp = fact;
	for(int i=0; i<=p; i++) tmp /= factorial(A[i]);
	ans += tmp;
}
// backtrack liet cac to hop co tong = len va hoan vi cua no
void backTrack(int p){
	for(int i=0; i<=len; i++){
		int tmp = sum, sml = A[p];
```

```
            A[p] += i; sum += i;
            if(p == 2 && sum == len) out(p);
            else if(p < 2 && sum <= len) backTrack(p+1);
            sum = tmp; A[p] = sml;
        }
}
int main(){
        cin.tie(0); ios_base::sync_with_stdio(false);
        int test; cin >> test;
        while(test--){
            ans = 0;
            cin >> n >> r >> b >> g;
            len = n - r - b - g;
            fact = factorial(n);
            A[0] = r; A[1] = b, A[2] = g;
            backTrack(0);
            cout << ans << endl;
        }
}
```