# 2025年中国矿业大学转计算机学院机试题面及题解

样例输入和输出记不太清了，希望后续补充。

以下仅为本人考时解法，并非正解。

## T1

### 题目描述

输入两个数 $m$ 和 $n$，输出区间 $[m,n]$ 中满足如下条件的所有数：

- 这个数是素数；
- 这个数的个位数字和十位数字的和的个位数等于百位数字。

输出要求从大到小，数之间空格隔开，并 5 个数字一行。

### AC Code

```cpp
#include <bits/stdc++.h>
using namespace std;

bool is_prime(int n)
{
    if(n == 1) return false;
    for(int i=2;i*i<=n;i++)
        if(n%i == 0) return false;
    return true;
}

bool is_valid(int n)
{
    int a = n%10;
    int b = n/10%10;
    int c = n/100%10;
    return (a+b)%10 == c;
}

int main()
{
    vector<int> res;
    int m,n;
    cin >> m >> n;
    for(int i=n;i>=m;i--)
        if(is_prime(i) && is_valid(i))
            res.push_back(i);
    for(int i=0;i<res.size();i++)
    {
        cout << res[i] << " ";
        if((i+1)%5 == 0) cout << endl;
    }
    return 0;
}
```

# T2

## 题目描述

对一个字符串，给定一个子串长度 $n$ ，做如下加密（以字符串 abcdefgh ，$n = 3$ 为例）：

- 子串 abc，索引为 0 ，向右移动一位变成 abc；
- 子串 def，索引为 1 ，向右移动一位变成 fde；
- 子串 gh，索引为 2 ，向右移动二位变成 gh。

把上述子串拼接，得到加密后的字符串 abcfdegh。

现在输入一个字符串，输出其加密后的字符串。

## AC Code

模拟即可，向右移位可以通过取余来实现。

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    string s,res = "";
    cin >> s;
    int len;
    cin >> len;
    for(int i=0;i*len<s.length();i++)
    {
        string t = "";
        for(int j=i*len;j<s.length() && j<=i*len+len-1;j++)
            t+=s[j];
        for(int j=0;j<t.length();j++)
            res+=t[(j-i+t.length())%t.length()];
    cout << res;
    return 0;
}
```

# T3

## 题目描述

计算从 1921年7月23日 到 2020年7月1日 之间的总分钟数。

## AC Code

答案是 52038720。计算 1922 年到 2020 年的天数，减去 22 即可。

```cpp
#include <bits/stdc++.h>
using namespace std;

bool is_run(int year)
{
    return (year%4==0 && year%100!=0) || year%400==0;
}

int main()
{
    int days = 0;
    for(int i=1922;i<=2020;i++)
    {
        if(is_run(i)) days+=366;
        else days+=365;
    }
    days-=22;
    cout << days*24*60;
    return 0;
}
```

# T4

第一行输入一个数 $t$ ，表示有 $t$ 个数，接下来 $t$ 行，输入一个数 $n$ ，将其分解成若干个斐波那契数的和的形式。如有多种表示方案，则输出斐波那契数最少的那一种；若还有多种，则输出后面的数大的方案。

## AC Code

我采用的是 DFS 回溯法。斐波那契数可以预先求出，接下来只需要寻找若干数相加等于 $n$ 的方案，这是一个典型的搜索问题。我采用了以下两个剪枝方法：

- 按照斐波那契数的顺序枚举。生成斐波那契数时已经按照顺序生成；
- 设定枚举起点。前面若被枚举过，则下一次递归从下一个数开始枚举，不需要回到前一个数。
- 当前的总和若大于 $n$ ，后续一定不能恰好凑成 $n$ ，直接回溯。

```cpp
#include <bits/stdc++.h>
using namespace std;

int n;
vector<int> fib,t;
vector<vector<int>> res;

bool cmp(vector<int> a,vector<int> b)
{
    if(a.size()!=b.size()) return a.size()<b.size();
    return a.back()>b.back();
}

void dfs(int sum,int cur)
{
    if(sum == n)
    {
        res.push_back(t);
        return ;
    }
    if(sum>n) return ;
    for(int i=cur;i<fib.size();i++)
    {
        t.push_back(fib[i]);
        dfs(sum+fib[i],i+1);;
        t.pop_back();
    }
}

int main()
{
    int T;
    cin >> T;
    while(T--)
    {
        fib.clear();
        res.clear();
        cin >> n;
        fib.push_back(1);
        fib.push_back(1);
        while(fib.back()<=n)
        {
            int p = fib[fib.size()-1]+fib[fib.size()-2];
            fib.push_back(p);
        }
        dfs(0,0);
        sort(res.begin(),res.end(),cmp);
        cout << n << "=";
        vector<int> ans = res[0];
        for(int i=0;i<ans.size()-1;i++) cout << ans[i] << "+";
        cout << ans.back() << endl;
    }
    return 0;
}
```

# T5

一个 $k$ 进制数被称为“波浪数”，需满足两个数字交替出现，如 12121、1A1A1A，ABABA 等。一个数称为 $k$ 重“波浪数”，需满足这个数的不同进制有 $k$ 个波浪数。需要注意：长度小于 3 的数一定不是“波浪数”。

现给出十进制数 $r_1, r_2, n_1, n_2, k$ ，输出进制范围 $[r_1, r_2]$ ，数字范围 $[n_!, n_2]$ 的 所有 $k$ 重波浪数，一行输出一个。

## AC Code

模拟即可，进制转换用取余，再枚举进制和数字。“波浪数”的判断只需判断奇数位和偶数位均相同即可。

```
#include <bits/stdc++.h>
using namespace std;

string str = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";

string to_k(int n,int k)
{
    string res = "";
    while(n)
    {
        res=str[n%k]+res;
        n/=k;
    }
    return res;
}

bool is_valid(string s)
{
    if(s.length()<3) return false;
    for(int i=0;i<s.length();i+=2)
        if(s[i]!=s[0]) return false;
    for(int i=1;i<s.length();i+=2)
        if(s[i]!=s[1]) return false;
    return true;
}

int main()
{
    int r1,r2,n1,n2,k;
    cin >> r1 >> r2 >> n1 >> n2 >> k;
    for(int i=n1;i<=n2;i++)
    {
        int sum=0;
        for(int j=r1;j<=r2;j++)
            if(is_valid(to_k(i,j))) sum++;
        if(sum>=k) cout << i << endl;
    }
    return 0;
}
```