```
}
void KMP(string text, string pattern)
{
   vector<int> lps = ConstructLPSarray(pattern);
   int j = 0, i = 0; // i=text, j=pattern
   int n = text.length();
   int m = pattern.length();
   bool ok = false;
   while (i < n)
   {
      if (text[i] == pattern[j])  i++, j++;
      else
      {
         if (j != 0)  j = lps[j - 1];
         else  i++;
      }
      if (j == m)
      {
         cout << i - m << endl;
         j = lps[j - 1];
         ok = true;
      }
   }
   if (!ok)
      cout << endl;
}
void solve()
{
   int n;
   while (cin >> n)
   {
      if (n == 0) break;
      string pattern, text;
      cin >> pattern >> text;
      KMP(text, pattern);
   }
}
```

Others:

Sorting pair Using Compare Function:

=>O(n*log(n))

If vector<pair<ll, ll>>**vec**{{3, 4}, {1, 2}, {3, 5}, {3, 2}, {6, 1}};

```
bool cmp(pair<ll, ll> a, pair<ll, ll> b)
{
   if (a.first != b.first) return a.first<b.first;
   //=>first value increasing order;
   return a.second>b.second;
   //=> second value descending order;
}
sort(vec.begin(), vec.end(), cmp);
//=>vec={{1,2}, {3,5}, {3,4}, {3,2}, {6,1}};
struct Node
{
   ll val, id, cost;
   bool operator<(const Node &rhs) const
   {
      //your main logic for the comparator goes here
      return make_pair(val, id) < make_pair(rhs.val, rhs.id);
   }
};
```

Minimum fraction:

If **a/b = c/d** => ex: 12/18 = 2/3

c = **a /__gcd(a,b)**; d = **b /_gcd(a,b);**

Find N'th Fibonacci number using Binet's Formula: =>O(1)

```
int fib(int n){
   double phi = (sqrt(5) + 1) / 2;
   return round(pow(phi, n) / sqrt(5));
}
```

Count words in a string using stringstream:

```
#include<sstream>
#include<string>
int countWords(string str)
{
stringstream sf(str);
   string word;
   int count = 0;
  while (sf >> word)
          count++; // <= you can change statement
   return count;
}
```

__int128 Data-type:

```
__int128 read()
{
   __int128 x = 0, f = 1;
   char ch = getchar();
   while (ch< '0' || ch> '9')
   {
      if (ch == '-') f = -1;
      ch = getchar();
   }
   while (ch>= '0' &&ch<= '9')
   {
      x = x * 10 + ch - '0';
      ch = getchar();
   }
   return x * f;
}
void print(__int128 x)
{
   if (x < 0) putchar('-'), x = -x;
   if (x > 9) print(x / 10);
```