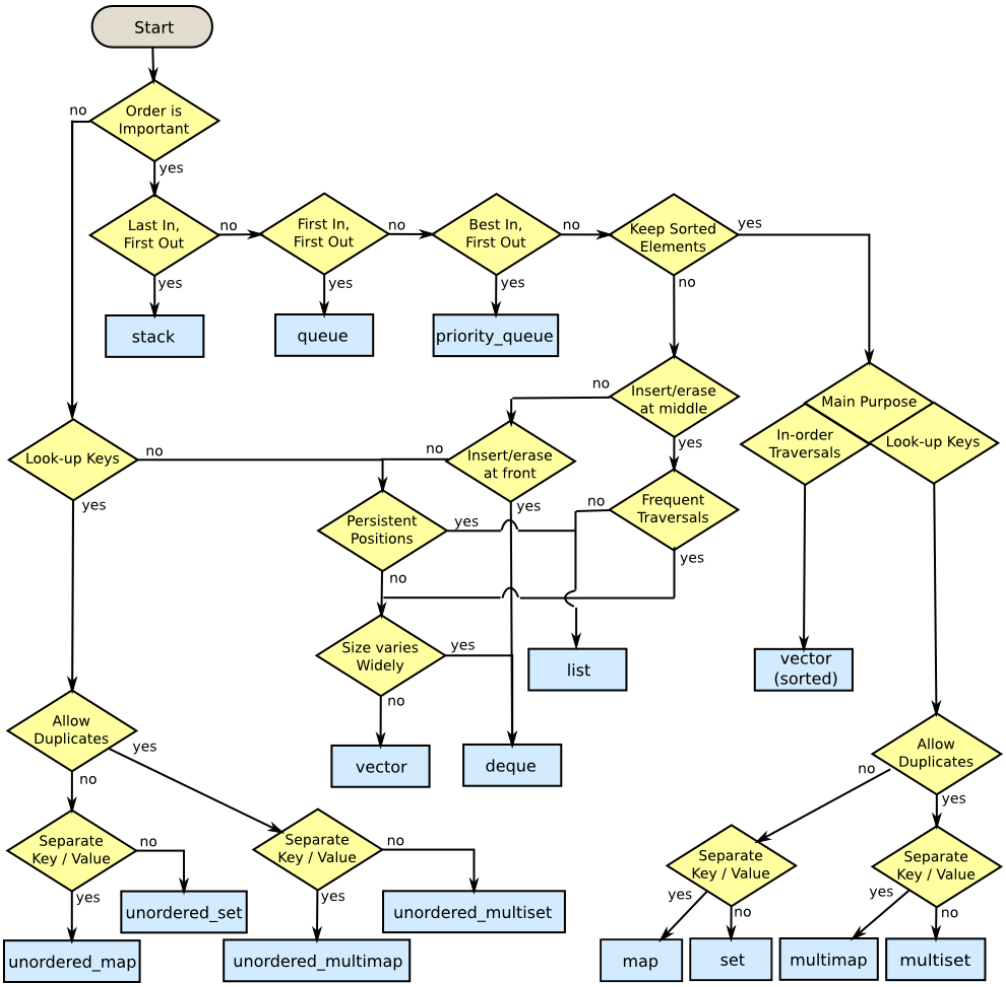
Template

```cpp
#pragma GCC optimization("Ofast")//-0.0+0.0=-0
#pragma GCC optimization("unroll-loops")
#pragma GCC target ("avx2")
#include<bits/stdc++.h>

using ull = unsigned long long;
using ll = long long;
using namespace std;
#define endl '\n'
#define dbg(...) cerr<<"LINE("<<__LINE__<<")->["<<#__VA_ARGS__<<"]:<<(__VA_ARGS__);
#define pb push_back
#define F first
#define S second

int main(){
ios_base::sync_with_stdio(0);cin.tie(0);
//freopen("in.txt", "r", stdin);
    int tc=2;
    //cin>>tc;
    int n;
    while(tc--){
        cin>>n;
        for(int i=0;i<n;++i){
            //code
        }
    }
return 0;
}
```

Containers



Data types

(-10^9 to +10^9)

int: -2,147,483,648 to 2,147,483,647
unsigned int: 0 to 4,294,967,295

(-10^18 to +10^18)

long long: -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
unsigned long long: 0 to 18,446,744,073,709,551,615

(7 digit precision):
float: 1,175494351 E - 38 to 3,402823466 E + 38

(15 digit precision):
double: 2,2250738585072014 E - 308 to 1,7976931348623158 E + 308

Misc Math functions

```
min({a, b, c, d});  max({a, b, c, d});
bool flag; cout << (flag ? "YES" : "NO");//ternary op
numeric_limits<unsigned int>::max();  numeric_limits<float>::min();
cout << fixed << setprecision(digits)<< var;// digits == 0 ? (round)
round(num);//1.45 -> 1 , 1.5 -> 2
trunc(num);//1.5 -> 1
ceil(num);//1.5 -> 2
floor(num);//1.5 -> 1
abs(num);// -1.5 -> 1.5, 1.5 -> 1.5
sqrt(num);  sqrtl(num);
pow(base, exp); powl(base,exp); pow(p, 1.0 / n);// nth root of p
var<<exp;//var*(2^exp)
var>>exp;//var/(2^exp)
```

First 25 primes

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

Algorithms

```
bool isPrime(int n){
    if (n<2) return false;
    if (n<=3) return true;
    if (!(n%2) || !(n%3)) return false;
    for (int i=5;i*i<=n;i+=6)
        if (!(n%i) || !(n%(i+2))) return false;
    return true;
}

// global constants for fibonnacci
const double sqrt5 = sqrt(5);
const double phi = (1 + sqrt5) / 2;
const double psi = (1 - sqrt5) / 2;
ull fibonacci(int n) {//O(1) of the first 70 fib numbers
    return round( (pow(phi, n) - pow(psi, n)) / (sqrt5) );
}
```

Ciruclar Array

```
int dist1 = abs(idx1-idx2);         int dist2 = n - dist1;
for (int i = idx1; i != idx2; i = (i - 1 + n) % n) {}//Left
for (int i = idx1; i != idx2; i = (i + 1 + n) % n) {}//right
```

sorting

```
sort(vec.begin(), vec.end());//ascending
sort(vec.begin(), vec.end(), greater<int>());//non ascending

inline bool myOrder(pair<int, int> p1, pair<int, int> p2) {
    return p1.first < p2.first;
}
vector<pair<int, int>> vec = {{3, 1}, {2, 5}, {1, 4}};
sort(vec.begin(), vec.end(), myOrder);
for(auto elem : vec)
    cout << "(" << elem.first << ", " << elem.second << ") ";
```

Standard Template Library (STL)

Iterators

```
for(vector<int>::iterator it=stl.begin(); it!=stl.end(); it++)
    cout<<*it<<" ";//(it) is memory pointer (*it) its value
for (const auto& pair : mapVar)
    cout << pair.first << ": " << pair.second;
```

Capacity:

```
stl.size(); //lenght of STL container
stl.empty(); // boolean function
```

Modifiers:

```
stl.clear();//clears the container but complexity O(n) better use swap(v,vv[i])
stl.erase(it);stl.erase(itBegin,itEnd);// removes elements from  range
stl.resize(size);stl.resize(size,initializer);stl.insert(it,elem)
stl.emplace_back(elem)//faster push_back (avoid copies), inmediate values
```

.fun()	.push_back(elem)	.pop_back()	.insert(elem)	.find(elem)
vector	x	x		find(v.beg(),v.end(),elem)
string	x	x	x	
maps			(make_pair(e,e))	x
sets			(elem)	x

Operations

```
if (stl.find(elem) != stl.end())//found
stl.count(elem);//count the occurrences of the element, sets and maps only
```

STL Algorithms

```
int sum = accumulate(v.begin(), v.end(), sumStart);
int product = accumulate(v.begin(), v.end(), 1,multiplies<int>());
merge(v1.begin(), v1.end(), v2.begin(), v2.end(), back_inserter(vecM));
fill(v.begin(), v.end(), value);iota(vec.begin(), vec.end(),start);
auto maxElement = max_element(vec.begin(), vec.end());
// use: cout << (maxElement != vec.end() ? *maxElement : "NO");
```

strings

```
string substr1=str1.substr(start, end);
int found1 = str.find(substr1);//first find at
int found2 = str.find(substr1, found1 + 1);//second find at
if (found1 != string::npos) //found

str1.replace(start, str2.size(), str2);
str1.erase(start, end);
str1.append(str2); str3 = str1 + str2;

char ch;
isalpha(ch);isdigit(ch);isupper(ch);islower(ch);//char bools
tolower(ch);toupper(ch);//alphabet
char numC='1'; int  numI=numC-'0';//toInt
int numI=1;   char numC=numI+'0';//toChar

std::stringstream ss;    ss << str1.substr(0, 1) << str1.substr(3, 4);

//circular string rotation
string s;
vector<string> v1(s.size());
vector<string> v2(s.size());
for(int i=0;i<s.size();i++){
    v1[i] = s.substr(i,s.size()-i) + s.substr(0,i);//left
    v2[i] = s.substr(s.size()-i,i) + s.substr(0,s.size()-i);//right
}
```