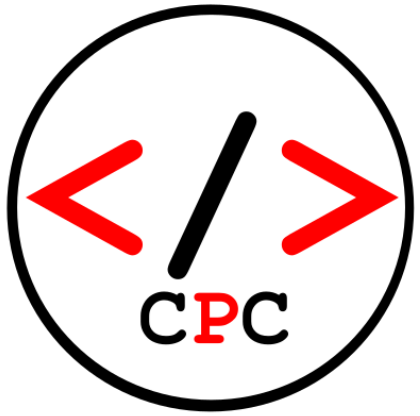


STL Part 2

Rosa A. S.

STL

- Aritmetika
- Bilangan Prima
- FPB – KPK
- Map
- Stack
- Queue
- Deque
- Vector 2 Dimensi = Vector di dalam Vector

- Website untuk CP - GeeksforGeeks

Aritmetika Modulo

5 mod 3 = 2
10 mod 2 = 0
21 mod 6 = 3

Sifat-sifat dasar dari operasi modulo adalah:

$(a+b) \bmod m = ((a \bmod m)+ (b \bmod m)) \bmod m$

$(a-b) \bmod m = ((a \bmod m)-(b \bmod m)) \bmod m$

$(a \times b) \bmod m = ((a \bmod m) \times (b \bmod m)) \bmod m$

$a^b \bmod m = (a \bmod m)^b \bmod m$

$(-a) \bmod m = (-(a \bmod m)+m) \bmod m$

$a/b \bmod m = a \times b^{-1} \bmod m$

Bilangan Prima

```
bool sieve(int n){
   int i, k;
   bool prime[n+1];
   memset(prime, true, sizeof(prime));
   prime[0]=false;
   prime[1]=false;
   ll m = sqrt(n);

   for (i=2; i<=m; i++)
      if (prime[i])
         for (k=i*i; k<=n; k+=i)
            prime[k]=false;

   return prime[n];
}
```

Bilangan Prima

- Kalau banyak yang harus dicek prima atau tidak
  - Buat array variable global sebanyak masukan terbesar – cek bisa tidak sebesar itu alokasi array-nya
  - Kemudian akses sesuai yang diperlukan

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

bool prime[1316134912];

void sieve(int n, int max){
   int i, k;
   ll m = sqrt(n);
   if(max < m){

      for (i=max; i<=m; i++)
          if (prime[i])
                for (k=i*i; k<=n; k+=i)
                      prime[k]=false;
   }
}
```

```
int main(){
   ios::sync_with_stdio(0);
   cin.tie(0);
   ll m;

   memset(prime, true, sizeof(prime));
   prime[0]=false;
   prime[1]=false;

   ll max = 2;

         while(cin >> m){
                if(max < m){
                      sieve(m);
                      max = m;
                }
                if(prime[m])cout << "prima\n";
                else cout << "bukan prima\n";
         }

         return 0;
}
```

FPB dan KPK

$$KPK(a,b) = \frac{a \times b}{FPB(a,b)}$$

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;


ll GCD_FPB(ll a, ll b){//euclid algorithm
   if (b==0) return a;
   return GCD_FPB(b,a%b);//rekursif
}

ll LCM_KPK(ll a, ll b){
   return b*a/GCD_FPB(a,b);
}
```

```
int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        ll m, n;

        cin >> m >> n;

        cout << GCD_FPB(m, n) << " - "
             << LCM_KPK(m, n) << "\n";
        return 0;
}
```

greatest common divisor (GCD)
lowest common multiple (LCM)

MAP

- sejenis set-nya pair
- seperti pair, namun hanya menyimpan elemen kunci yang unik dan terurut berdasarkan nilai kuncinya

MAP

1 aku
2 adalah
5 anak
6 gembala
4 selalu
3 riang
67 serta
12 gembira
23 karena
6 aku
6 senang
6 bekerja
2 tak
2 pernah
12 malas
12 ataupun
12 lelah

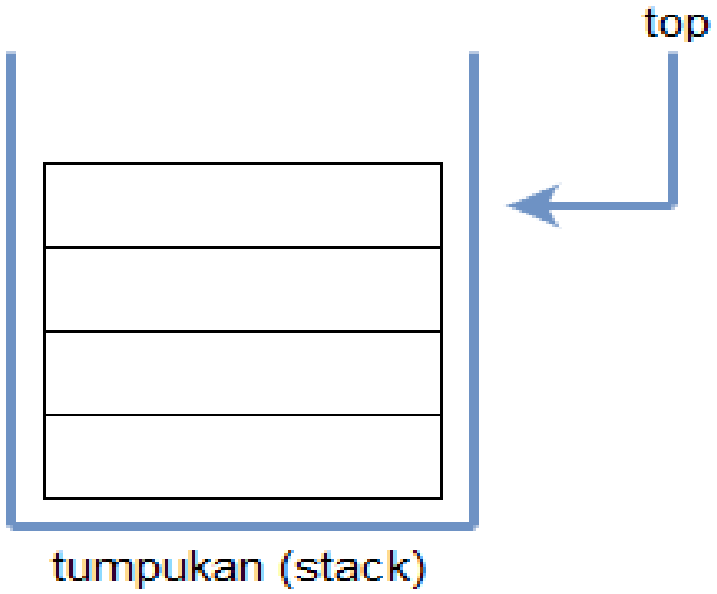
1 aku
2 adalah
3 riang
4 selalu
5 anak
6 gembala
12 gembira
23 karena
67 serta

MAP

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        map<int, string> m;

        int a,i;
        string str;

        while(cin >> a >> str)
                m.insert(pair<int, string>(a, str));

        map<int,string>::const_iterator it;

        for (it = m.begin();it!=m.end();it++)
                cout << it->first << " - " << it->second << "\n";

        return 0;
}
```

Stack - Tumpukan

Tumpukan atau *stack* adalah salah satu konsep struktur data yang memiliki sistem kerja yang terakhir masuk adalah yang pertama keluar (LIFO = *Last In First Out*)

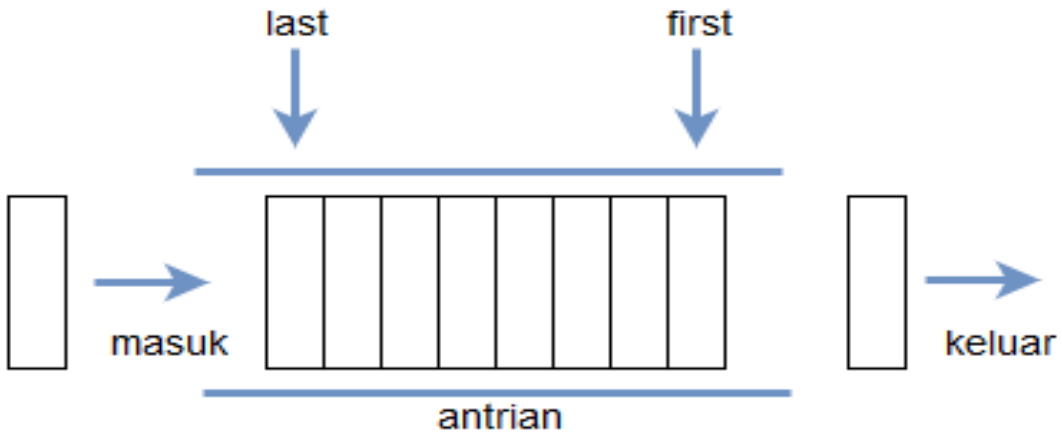


Stack

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        stack<ll> s;
        ll a;

        while(cin >> a)s.push(a);

        while (!s.empty()){
            cout << s.top() << "\n";
            s.pop();
        }

        return 0;
}
```

Queue - Antrian

- Antrian atau queue (baca : qyu) adalah salah satu konsep struktur data yang memiliki sistem kerja pertama masuk maka akan menjadi yang pertama keluar (FIFO = *First In First Out*) seperti halnya antrian yang ada pada dunia nyata



Queue

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        queue<ll> q;
        ll a;

        while(cin >> a)q.push(a);

       cout << "size: " << q.size() << "\n";
       cout << "front: " << q.front() << "\n";
       cout << "back: " << q.back() << "\n";

       while (!q.empty()){
         cout << q.front() << "\n";
         q.pop();
       }

       return 0;
}
```

Deque

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        deque<ll> dq;
        ll a;
        int i = 1;
        while(cin >> a)
                if(i%2==1)dq.push_front(a);
                else dq.push_back(a);

         cout << "size: " << dq.size() << "\n";
         cout << "front: " << dq.front() << "\n";
         cout << "back: " << dq.back() << "\n";

        deque<ll>::const_iterator it;

        for(it=dq.begin();it!=dq.end();it++){
            cout << *it << "\n";
        }

        return 0;
}
```

Vector 2D

```
#include <bits/stdc++.h>

using namespace std;

typedef long long ll;

int main(){
        ios::sync_with_stdio(0);
        cin.tie(0);

        vector<vector<ll>> v;
        string s;

        while(getline(cin, s)){
                stringstream ss(s);
                string word;
                vector<ll> pv;
                while (ss >> word) {
                        pv.push_back(stoll(word));
                }
                v.push_back(pv);
        }
        int i,j;
        for(i=0;i<v.size();i++){
                for(j=0;j<v[i].size();j++)
                        cout << v[i][j] << " ";
                cout << "\n";
        }

        return 0;
}
```

Next:

- Part of Math
- Big-O
- Brute Force
- Divide and Conquer