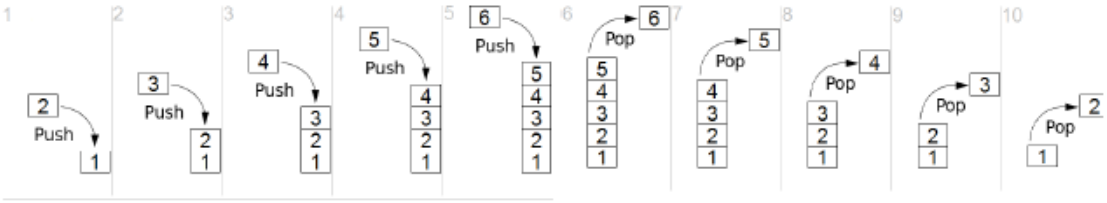
BAB 5 . STACK

5.1 Definisi Stack

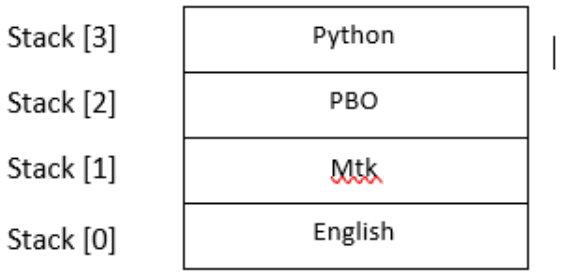
Stack merupakan tumpukan dari benda. Konsep utama dari stack adalah LIFO (Last In First Out). Yang artinya benda atau data yang dimasukkan diurutan terakhir, menjadi data atau benda yang akan dikeluarkan pertama kali dalam stack.
Stack memiliki batas maksimal banyak data dan stack bisa diimplementasikan dengan array dan linked list



Operasi standar pada stack :

- Push = Menambahkan elemen keatas stack (tumpukan)
- Pop = Menghapus satu data ditumpukan teratas
- isFull = Memeriksa apakah ruang stack sudah penuh atau belum
- isEmpty = Memeriksa apakah ruang stack kosong atau tidak
- Peek = Melihat atau mengintip data diposisi tertentu
- Count = Menghitung banyak data pada stack
- Change = Mengubah data diposisi tertentu
- Display = Mencetak semua data pada stack
- Destroy = Menghapus atau membersihkan semua data pada stack

5.2 Stack Dengan Array



Pada ilustrasi diatas terdapat sebuah array yang berukuran 4. Terdapat variable top yang berada pada indeks ke 3 dengan nilai python. Variable top ini menunjukan posisi variable teratas pada stack. Dimana nilai pada indeks top merupakan data terakhir dipush dan apabila ingin melakukan pop, maka nilai 2 adalah nilai yang pertama keluar.

Program push (insert) pada array

```cpp
#include <iostream>

using namespace std;

int maks=5,top = 0;
string buku[5];

void PushBuku(string data){

    if(top>=maks){
        cout<<"Data Full"<<endl;
    }else{
        buku[top]=data;
        top++;
    }
}

void DisplayBuku(){
    cout<<"Data Stack Buku"<<endl;
    cout<<"==============="<<endl;
    for(int i = maks-1 ; i>=0;i--){
            if(buku[i] != ""){
            cout<<"Stack ke -"<<i<<" = "<<buku[i]<<endl;
            }
        }
    }

int main(){
    PushBuku("Sherlock");
    PushBuku("Hujan");
    DisplayBuku();
    cout<<endl;

    PushBuku("Negeri 5 Menara");
    PushBuku("Senja");
    PushBuku("Sejarah Dunia");
    DisplayBuku();
    cout<<endl;

    PushBuku("Kata");
    DisplayBuku();
return 0;
}
```

Output :

```
Data Stack Buku
===============
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Stack Buku
===============
Stack ke -4 = Sejarah Dunia
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Full
Data Stack Buku
===============
Stack ke -4 = Sejarah Dunia
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock
```

Program cek apakah stuck pada array sudah penuh atau kosong

```
//Cek Apakah Stack Penuh
bool isFull(){
    if(top==maks){
        cout<<"Stack Sudah Penuh ";
        return true;
    }else{
        cout<<"Stack Masih Kosong "<<(maks-top)<<" Data";
        return false;
    }
}

//Cek Apakah Stack Kosong
bool isEmpty(){
    if(top==0){
        return true;
    }else{
        return false;
    }
}
```

Program pop(delete) pada array

```
//Menghapus Data Teratas
void PopBuku(){
    if(isEmpty()){
        cout<<"Data Kosong"<<endl;
    }else{
     buku[top-1]= "";
    }
}
```

Memanggil fungsi pop

```
int main(){
    PushBuku("Sherlock");
    PushBuku("Hujan");
    DisplayBuku();
    cout<<endl;

    PushBuku("Negeri 5 Menara");
    PushBuku("Senja");
    DisplayBuku();
    cout<<endl;

    PushBuku("Kata");
    DisplayBuku();

    isFull();
    PopBuku();
    DisplayBuku();
return 0;
}
```

Output:

```
Data Stack Buku
===============
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Stack Buku
===============
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Stack Buku
===============
Stack ke -4 = Kata
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock
Stack Sudah Penuh

Data Stack Buku
===============
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock
```

Program peek pada array

```
//Melihat index Data
void PeekBuku(int posisi){
    if(isEmpty()){
        cout<<"Data Kosong"<<endl;
    }else{
        int index = top-1;
        for(int i=0;i<posisi;i++){
            index--;
        }
        cout<<index<<endl;
    }
}
```

Memanggil fungsi peek

```
int main(){
    PushBuku("Sherlock");
    PushBuku("Hujan");
    DisplayBuku();
    cout<<endl;

    PushBuku("Negeri 5 Menara");
    PushBuku("Senja");
    DisplayBuku();
    cout<<endl;

    PushBuku("Kata");
    DisplayBuku();

    isFull();
    PopBuku();
    DisplayBuku();
    PeekBuku(1);
return 0;
}
```

Output

```
Data Stack Buku
===============
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Stack Buku
===============
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock

Data Stack Buku
===============
Stack ke -4 = Kata
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock
Stack Sudah Penuh

Data Stack Buku
===============
Stack ke -3 = Senja
Stack ke -2 = Negeri 5 Menara
Stack ke -1 = Hujan
Stack ke -0 = Sherlock
3
```

5.3 Stack dengan linked list

Operasi untuk stack dengan linked list :

isEmpty fungsi memeriksa apakah stack yang ada masih kosong

Push fungsi memasukkan elemen bar uke dalam stack. Push disini mirip dengan insert dalam single linked list biasa.

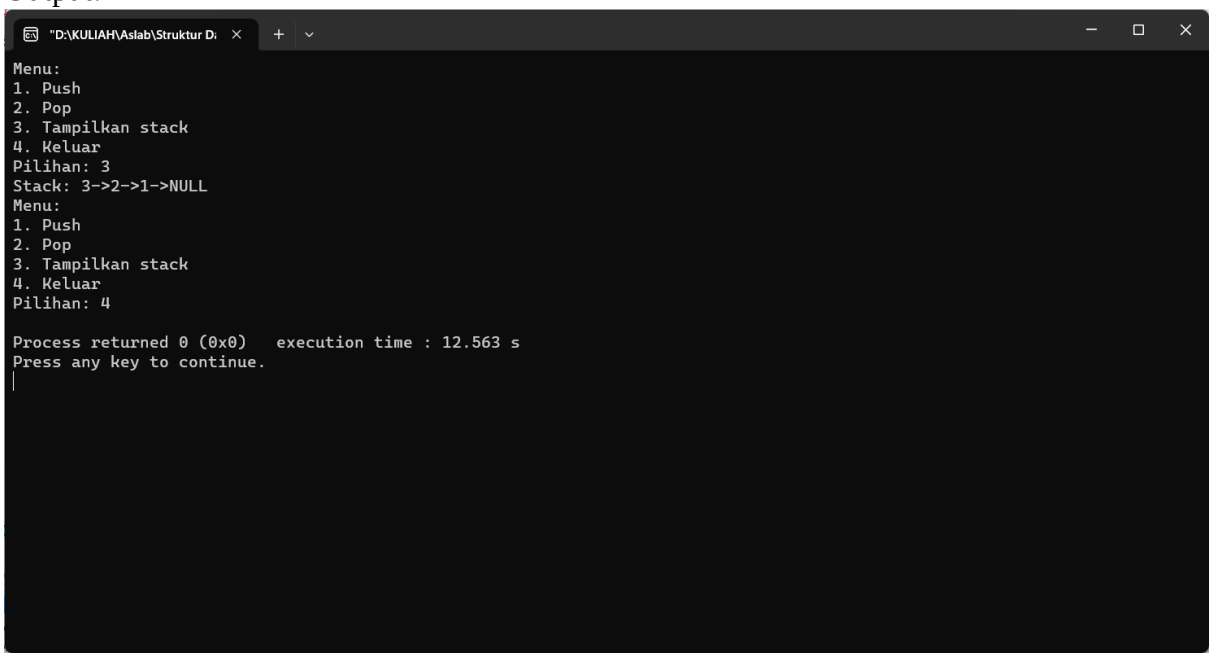
Pop fungsi ini mengeluarkan elemen teratas stack

Contoh Program

```
1   #include <iostream>
2   #include <windows.h>
3   using namespace std;
4
5   struct Node {
6       int data;
7       Node* next;
8   };
9
10  struct Stack {
11      Node* top;
12      Stack() {
13          top = NULL;
14      }
15
16      void push(int value) {
17          Node* newNode = new Node;
18          newNode->data = value;
19          newNode->next = top;
20          top = newNode;
21          cout << "Nilai " << value << " sudah diinput ke dalam stack.\n";
22          system("cls");
23      }
24
```

```
25      void pop() {
26          if (top == NULL) {
27              cout << "Stack kosong.\n";
28              return;
29          }
30          Node* temp = top;
31          int poppedValue = top->data;
32          top = top->next;
33          delete temp;
34          cout << "Nilai " << poppedValue << " sudah dihapus dari stack.\n";
35          system("cls");
36      }
37
38      void display() {
39          if (top == NULL) {
40              cout << "Stack kosong.\n";
41              return;
42          }
43          Node* currentNode = top;
44          cout << "Stack: ";
45          while (currentNode != NULL) {
46              cout << currentNode->data << "->";
47              currentNode = currentNode->next;
48          }
49          cout<< "NULL" << endl;
50      }
51  };
52
53  int main() {
54      Stack s;
55      int pilih, nilai;
56      while (true) {
57          cout << "Menu:\n";
58          cout << "1. Push\n";
59          cout << "2. Pop\n";
60          cout << "3. Tampilkan stack\n";
61          cout << "4. Keluar\n";
62          cout << "Pilihan: ";
63          cin >> pilih;
64          switch (pilih) {
65              case 1:
66                  cout << "Masukkan nilai yang ingin di-push: ";
67                  cin >> nilai;
68                  s.push(nilai);
69                  break;
70              case 2:
71                  s.pop();
72                  break;
73              case 3:
74                  s.display();
75                  break;
76              case 4:
77                  return 0;
78              default:
79                  cout << "Pilihan tidak valid.\n";
80                  break;
81          }
82      }
83  }
84
```

Output:



LATIHAN

Buatlah program pemecahan masalah Menara Hanoi yang memindahkan lempengan dari menara A ke menara C dengan perantara menara B dengan jumlah data 3 (10,20,30). Buat program tersebut dengan menerapkan konsep struktur data stack(boleh dengan stack array atau stack linked list), tapi tidak boleh menggunakan library stack.

Contoh Output Program:

1. Pindahkan batu 10 dari A ke C
2. Pindahkan batu 20 dari A ke B
3. Pindahkan batu 10 dari C ke B
4. Pindahkan batu 30 dari A ke C
5. Pindahkan batu 10 dari B ke A
6. Pindahkan batu 20 dari B ke C
7. Pindahkan batu 10 dari A ke C