

LAPORAN PRAKTIKUM

▪ Identitas Praktikum

Nama MK : Struktur Data
Kode MK : CCK2AAB4
Bobot SKS : 4 SKS
Tempat : L-Program, Gedung DC, lantai 3
Hari, tanggal : Selasa, 5 November 2024
Jam : 12:30-15:30 WIB
Topik praktikum : Modul-7 STACK

▪ Identitas Mahasiswa

Nama lengkap : Afad Fath Musyarof Halim
NIM : 2211104030
Program Studi : S-1 Software Engineering

▪ Hasil Praktikum

7.1. STACK

7.1.1. isFull

```
bool isFull() { return top == MAX - 1; }
```

- Mengembalikan True apabila stack penuh

7.1.2. isEmpty

```
bool isEmpty() { return top == -1; }
```

- Mengembalikan true apabila stack kosong

7.1.3. Push

```
void push(int x){
    if (isFull())
    {
        std::cout << "Stack sudah Penuh\n";
        return;
    }
    arr[++top] = x;
}
```

- Jika penuh maka menghentikan fungsi
- Menambahkan elemen di urutan atas stack
- Contoh:

- o Kode:

```
std::cout << "Element: ";
s.display();

s.push(10);
s.push(20);
s.push(30);

std::cout << "Element: ";
s.display();
```

- o Output:

```
● Afadfath | output main
# & .\'main.exe\'
Element: Stack is empty
Element: 30 20 10
○ Afadfath | output main
# []
```

7.1.4. Pop

```
void pop(){
    if (isEmpty()){
        std::cout << "Stack Kosong\n";
        return;
    }
    top--;
}
```

- Jika stack kosong maka menghentikan operasi
- Menghapus elemen dari urutan stack paling atas
- Contoh
 - o Kode:

```
s.push(10);
s.push(20);
s.push(30);
std::cout << "Element (Sebelum Pop): ";
s.display();
s.pop();
std::cout << "Elemen (Setelah Pop): ";
s.display();
```

- o Output:

```
Afadfath | output main
# & .\main.exe
Element (Sebelum Pop): 30 20 10
Elemen (Setelah Pop): 20 10
Afadfath | output main
# []
```

7.1.5. Peek

```
void peek() {
    if (isEmpty()) {
        std::cout << "Stack kosong\n";
        return;
    }

    std::cout << arr[top] << "\n";
}
```

- Menghentikan fungsi apabila stack kosong
- Menampilkan Elemen stack paling atas
- Contoh:

- Kode

```
s.push(10);
s.push(20);
s.push(30);

std::cout << "Element: ";
s.display();
std::cout << "Element (Top): ";
s.peak();
```

- Output:

```
Afadfath | output main
# & .\main.exe
Element: 30 20 10
Element (Top): 30
Afadfath | output main
#
```

7.1.6. Display

```
void display(){
    if (isEmpty()){
        std::cout << "Stack kosong\n";
        return;
    }

    for (int i = top; i >= 0; i--){
        std::cout << arr[i] << " ";
    }

    std::cout << "\n";
}
```

- Menghentikan fungsi jika stack kosong
- Menampilkan semua elemen yang ada pada stack dengan mengiterasikan semua elemen dari atas/top stack

7.2. Latihan

1. Buat ADT stack

```
Type infotype : integer
Type Stack <
  info : array [20] of integer
  top : integer
>
prosedur CreateStack( in/out S : Stack )
prosedur push( in/out S : Stack, in x : infotype)
fungsi pop( in/out S : Stack ) : infotype
prosedur printInfo( in S : Stack )
prosedur balikStack( in/out S : Stack )
```

Program 2 Stack.h

Buatlah implementasi ADT Stack menggunakan Array pada file "stack.cpp" dan "main.cpp"

```
int main()
{
    cout << "Hello world!" <<
endl;
    Stack S;
    createStack(S);
    Push(S,3);
    Push(S,4);
    Push(S,8);
    pop(S);
    Push(S,2);
    Push(S,3);
    pop(S);
    Push(S,9);
    printInfo(S);
    cout<<"balik stack"<<endl;
    balikStack(S);
    printInfo(S);
    return 0;
}
```

```
Hello world!
[TOP] 9 2 4 3
balik stack
[TOP] 3 4 2 9
```

Gambar 7-11 Output stack

Gambar 7-12 Main stack

- Source Code:

o Stack.h

```
#ifndef STACK_H
#define STACK_H

const int MAX = 20;
typedef int infotype;

struct Stack {
    infotype info[MAX];
    int top;
};

void CreateStack(Stack &S);
void push(Stack &S, infotype x);
infotype pop(Stack &S);
void printInfo(const Stack &S);
void balikStack(Stack &S);

#endif // STACK_H
```

- Stack.cpp

```
#include "stack.h"
#include <iostream>
using namespace std;

void CreateStack(Stack &S){ S.top = -1; };

void push(Stack &S, infotype x){
    if (S.top == MAX - 1){ cout << "Stack sudah penuh\n"; return;}

    S.info[++S.top] = x;
};

infotype pop(Stack &S){
    if (S.top == -1){ cout << "Stack kosong\n"; return -1;}

    return S.info[S.top--];
};

void printInfo(const Stack &S){
    if (S.top == -1){ cout << "Stack kosong\n"; return;}

    cout << "[TOP] ";
    for (int i = S.top; i >= 0; i--){
        cout << S.info[i] << " ";
    }

    cout << "\n";
};

void balikStack(Stack &S){
    Stack temp;
    CreateStack(temp);

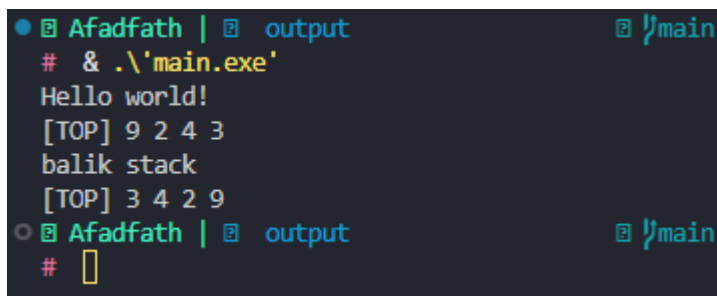
    while (S.top != -1){
        push(temp, pop(S));
    }

    S = temp;
};
```

- Main.cpp

```
int main() {  
    cout << "Hello world!" << endl;  
  
    Stack S;  
    CreateStack(S);  
  
    push(S, 3);  
    push(S, 4);  
    push(S, 8);  
  
    pop(S);  
  
    push(S, 2);  
    push(S, 3);  
  
    pop(S);  
    push(S, 9);  
    printInfo(S);  
  
    cout << "balik stack" << endl;  
    balikStack(S);  
    printInfo(S);  
  
    return 0;  
}
```

- Output:



```
● Afadfath | output main  
# & .\'main.exe'  
Hello world!  
[TOP] 9 2 4 3  
balik stack  
[TOP] 3 4 2 9  
○ Afadfath | output main  
#
```

- Penjelasan:

- `CreateStack` untuk membuat stack kosong
- `push` untuk menambahkan elemen di urutan atas stack
- `pop` untuk menghapus elemen dari urutan atas stack
- `printInfo` untuk menampilkan isi dari stack dari atas ke bawah
- `balikStack` untuk membalikkan urutan elemen di stack

2. Tambahkan prosedur **pushAscending(in/out S : Stack, in x : integer)**

```
int main()
{
    cout << "Hello world!" << endl;
    Stack S;
    createStack(S);
    pushAscending(S,3);
    pushAscending(S,4);
    pushAscending(S,8);
    pushAscending(S,2);
    pushAscending(S,3);
    pushAscending(S,9);
    printInfo(S);
    cout<<"balik stack"<<endl;
    balikStack(S);
    printInfo(S);
    return 0;
}
```

```
Hello world!
[TOP] 9 8 4 3 3 2
balik stack
[TOP] 2 3 3 4 8 9
```

Gambar 7-13 Output stack push ascending

Gambar 7-14 Main stack dengan push ascending

- Source Code:

- o Tambahan stack.h

```
void pushAscending(Stack &S, int x);
```

- o Tambahan stack.cpp

```
void pushAscending(Stack &S, int x) {
    Stack temp;
    CreateStack(temp);

    while (S.top != -1 && S.info[S.top] > x) {
        push(temp, pop(S));
    }

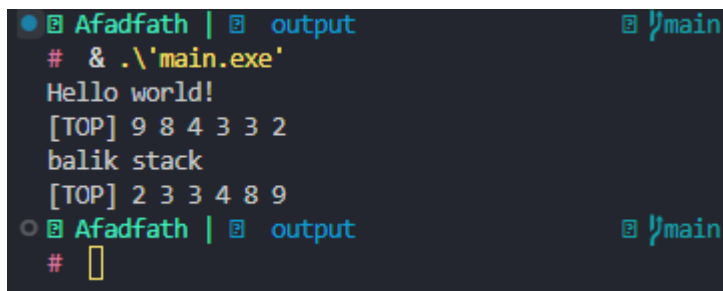
    push(S, x);

    while (temp.top != -1) {
        push(S, pop(temp));
    }
}
```


- o main.cpp

```
int main() {  
    cout << "Hello world!" << endl;  
    Stack S;  
    CreateStack(S);  
  
    pushAscending(S,3);  
    pushAscending(S,4);  
    pushAscending(S,8);  
    pushAscending(S,2);  
    pushAscending(S,3);  
    pushAscending(S,9);  
  
    printInfo(S);  
  
    cout<<"balik stack"<<endl;  
    balikStack(S);  
    printInfo(S);  
  
    return 0;  
}
```

- Output:



```
Afadfath | output main  
# & .\'main.exe'  
Hello world!  
[TOP] 9 8 4 3 3 2  
balik stack  
[TOP] 2 3 3 4 8 9  
Afadfath | output main  
# []
```

- Penjelasan:

- o membuat stack sementara (temp)
- o Memindah elemen pada stack ke temp selama elemen tersebut lebih besar dari input
- o Memasukkan inputan ke stack
- o Mengembalikan elemen yang ada pada temp ke stack dengan urutan setelah inputan yang telah di masukan ke stack

3. Tambahkan prosedur `getInputStream(in/out S : Stack)`. Prosedur akan terus membaca dan menerima input user dan memasukkan setiap input ke dalam stack hingga user menekan tombol enter. Contoh: gunakan `cin.get()` untuk mendapatkan inputan user.

```
int main()
{
    cout << "Hello world!" << endl;
    Stack S;
    createStack(S);
    getInputStream(S);
    printInfo(S);
    cout<<"balik stack"<<endl;
    balikStack(S);
    printInfo(S);
    return 0;
}
```

Gambar 7-16 Main stack dengan *input stream*

```
Hello world!
4729601
[TOP] 1 0 6 9 2 7 4
balik stack
[TOP] 4 7 2 9 6 0 1
```

Gambar 7-15 *Output stack*

- Source Code:

- o Tambahkan `stack.h`

```
void getInputStream(Stack &S);
```

- o Tambahkan `stack.cpp`

```
void getInputStream(Stack &S) {
    char inputan;

    while (true) {
        inputan = cin.get();

        if (isdigit(inputan)) {
            if (S.top == MAX - 1) {
                cout << "Stack sudah penuh\n";
                break;
            }

            push(S, inputan - '0');
        }

        if (inputan == '\n') {
            break;
        }
    }
}
```

- Main.cpp

```
int main() {  
    cout << "Hello world!" << endl;  
    Stack S;  
    CreateStack(S);  
  
    getInputStream(S);  
    printInfo(S);  
  
    cout<<"balik stack"<<endl;  
    balikStack(S);  
    printInfo(S);  
  
    return 0;  
}
```

- Output:

```
● Afadfath | output 1/1 main  
# & .\main.exe  
Hello world!  
12345  
[TOP] 5 4 3 2 1  
balik stack  
[TOP] 1 2 3 4 5  
○ Afadfath | output 1/1 main  
#
```

- Penjelasan:

- Inputan = cin.get() untuk menyimpan inputan dari user berdasarkan setiap karakter
- Jika inputan adalah digit (0-9) maka akan di masukkan ke stack apabila belum penuh, jika sudah penuh maka fungsi berhenti
- Jika user menekan enter (newline) maka fungsi berhenti