

**LAPORAN PRAKTIKUM STRUKTUR
DATA**

**MODUL VII
STACK**



Disusun Oleh :

**NAMA : ADIKA AUNURFIKRI NOVIYANTO
NIM : 103112400195**

Dosen

FAHRUDIN MUKTI WIBOWO

**PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025**

A. Dasar Teori

C++ adalah pengembangan dari bahasa c yang dibuat oleh Bjarne Stroustrup sekitar tahun 1980-an. C++ disebut bahasa multi-paradigma, artinya bisa dipakai dengan gaya prosedural (pakai fungsi biasa), berorientasi objek (pakai class dan object), atau bahkan gabungan keduanya. C++ punya dasar-dasar seperti variabel, operator percabangan (if, switch), perulangan (for, while), dan bisa memakai class untuk membuat objek.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1

stack.h

```
#ifndef STACK_H_INCLUDED
#define STACK_H_INCLUDED

typedef int infotype;

struct Stack {
    infotype info[20];
    int top;
};

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

#endif
```

stack.cpp

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

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

void push(Stack &S, infotype x) {
    if (S.top < 19) {
        S.top++;
        S.info[S.top] = x;
    }
}

infotype pop(Stack &S) {
    if (S.top >= 0) {
        infotype x = S.info[S.top];
        S.top--;
        return x;
    }
    return -1;
}

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

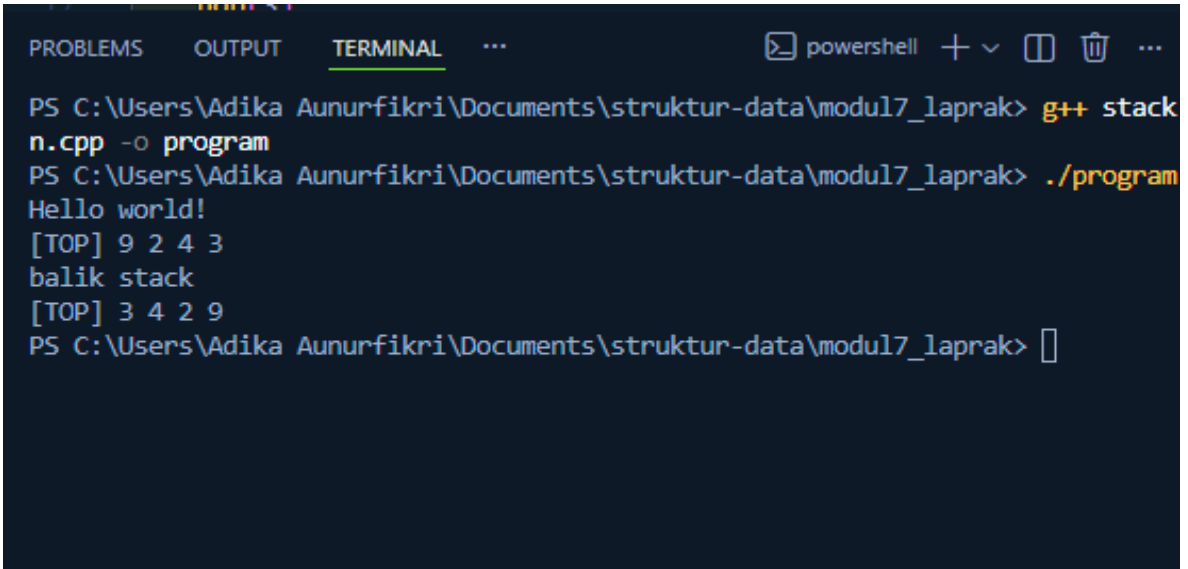
void balikStack(Stack &S) {
    Stack T;
```

```
createStack(T);  
while (S.top >= 0) {  
    push(T, pop(S));  
}  
S = T;  
}
```

Main.cpp

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

Screenshots Output:



```
PROBLEMS OUTPUT TERMINAL ... powershell + v [ ] [ ] ...
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> g++ stack
n.cpp -o program
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> ./program
Hello world!
[TOP] 9 2 4 3
balik stack
[TOP] 3 4 2 9
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> [ ]
```

Guided 2

stack.h

```
#ifndef STACK_H_INCLUDED
#define STACK_H_INCLUDED

typedef int infotype;

struct Stack {
    infotype info[20];
    int top;
};

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

void pushAscending(Stack &S, infotype x);

#endif
```

stack.cpp

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

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

void push(Stack &S, infotype x) {
    if (S.top < 19) {
        S.top++;
        S.info[S.top] = x;
    }
}

infotype pop(Stack &S) {
    if (S.top >= 0) {
        infotype x = S.info[S.top];
        S.top--;
        return x;
    }
    return -1;
}

void printInfo(Stack S) {
    cout << "[TOP] ";
    for (int i = S.top; i >= 0; i--) {
        if (S.info[i] >= 48 && S.info[i] <= 57) {
            cout << (char)S.info[i] << " ";
        } else {
            cout << S.info[i] << " ";
        }
    }
    cout << endl;
```

```

}

void balikStack(Stack &S) {
    Stack T;
    createStack(T);
    while (S.top >= 0) {
        push(T, pop(S));
    }
    S = T;
}

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

    while (S.top >= 0 && S.info[S.top] > x) {
        push(temp, pop(S));
    }

    push(S, x);

    while (temp.top >= 0) {
        push(S, pop(temp));
    }
}

```

Main.cpp

```

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

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);

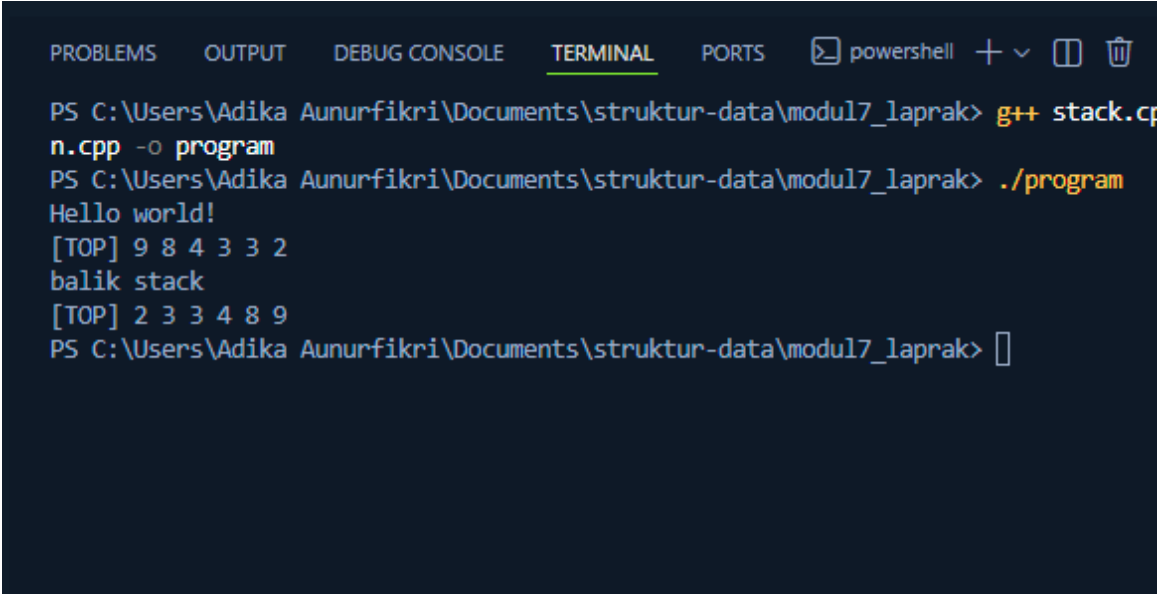
printInfo(S);

cout << "balik stack" << endl;
balikStack(S);

printInfo(S);

return 0;
}
```

Screenshots Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  powershell + v [] []
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> g++ stack.cpp
n.cpp -o program
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> ./program
Hello world!
[TOP] 9 8 4 3 3 2
balik stack
[TOP] 2 3 3 4 8 9
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> []
```


Guided 3

stack.h

```
#ifndef STACK_H_INCLUDED
#define STACK_H_INCLUDED

typedef int infotype;

struct Stack {
    infotype info[20];
    int top;
};

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

void pushAscending(Stack &S, infotype x);

void getInputStream(Stack &S);

#endif
```

stack.cpp

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

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

void push(Stack &S, infotype x) {
    if (S.top < 19) {
        S.top++;
        S.info[S.top] = x;
    }
}
```

```

infotype pop(Stack &S) {
    if (S.top >= 0) {
        infotype x = S.info[S.top];
        S.top--;
        return x;
    }
    return -1;
}

void printInfo(Stack S) {
    cout << "[TOP] ";
    for (int i = S.top; i >= 0; i--) {
        if (S.info[i] >= 48 && S.info[i] <= 57) {
            cout << (char)S.info[i] << " ";
        } else {
            cout << S.info[i] << " ";
        }
    }
    cout << endl;
}

void balikStack(Stack &S) {
    Stack T;
    createStack(T);
    while (S.top >= 0) {
        push(T, pop(S));
    }
    S = T;
}

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

    while (S.top >= 0 && S.info[S.top] > x) {

```

```

        push(temp, pop(S));
    }

    push(S, x);

    while (temp.top >= 0) {
        push(S, pop(temp));
    }
}

void getInputStream(Stack &S) {
    char c;

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

        if (c == '\n') {
            break;
        }

        push(S, (int)c);
    }
}

```

Main.cpp

```

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

int main() {
    cout << "Hello world!" << endl;

    Stack S;
    createStack(S);
}

```

```

    getInputStream(S);
    printInfo(S);

    cout << "balik stack" << endl;
    balikStack(S);

    printInfo(S);

    return 0;
}

```

Screenshots Output:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + v [] [X]
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> g++ stack.cpp n.cpp -o program
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak> ./program
Hello world!
4729601
[TOP] 1 0 6 9 2 7 4
balik stack
[TOP] 4 7 2 9 6 0 1
PS C:\Users\Adika Aunurfikri\Documents\struktur-data\modul7_laprak>

```

Deskripsi:

Pada tugas ini mahasiswa diminta mengimplementasikan struktur data Stack menggunakan array dengan membuat operasi dasar seperti createStack, push, pop, printInfo, dan balikStack untuk memahami prinsip LIFO. Selanjutnya, mahasiswa mengembangkan operasi push menjadi pushAscending yang memastikan elemen yang dimasukkan tetap tersusun ascending dari bawah ke atas melalui penggunaan stack sementara. Terakhir, fungsi getInputStream ditambahkan untuk membaca input karakter per karakter menggunakan cin.get() hingga user menekan ENTER, lalu menyimpannya dalam stack dan menampilkannya kembali. Ketiga soal ini memberikan pemahaman menyeluruh tentang bagaimana stack bekerja, bagaimana operasi dasar dapat dimodifikasi, dan bagaimana stack dapat digunakan untuk memproses data numerik maupun karakter.

C. Kesimpulan

Dari seluruh rangkaian tugas dapat disimpulkan bahwa struktur data Stack merupakan struktur sederhana namun sangat fleksibel, karena selain mampu melakukan operasi dasar LIFO, juga dapat dikembangkan menjadi operasi tambahan seperti pushAscending dan pemrosesan input karakter dengan getInputStream tanpa meninggalkan konsep utamanya. Implementasi menggunakan array menunjukkan bahwa pengelolaan stack dapat dilakukan dengan efisien hanya melalui manipulasi indeks, sementara pengembangan operasi membuktikan bahwa stack dapat disesuaikan untuk berbagai kebutuhan seperti penyusunan data terurut maupun pembacaan stream karakter. Secara keseluruhan, praktikum ini memperkuat pemahaman mengenai cara kerja stack, pemanfaatannya, serta potensinya untuk dikembangkan menjadi fungsi yang lebih kompleks.

D. Referensi

- W3Resource. (2020). *C++ String Exercises: Convert digit/number to words*.
- GeeksforGeeks. (2020). *Loops in C++ (for, while, do-while)*.