

LAPORAN PRAKTIKUM

**Modul 7
“STACK”**



Disusun Oleh:

**Benedictus Qsota Noventino Baru - 2311104029
S1SE07A**

Dosen :

Yudha Islami Sulistya, S.Kom., M.Cs

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM
PURWOKERTO
2024**

LATIHAN MODUL

Soal 1

File stack.h

```
1  #ifndef STACK_H
2  #define STACK_H
3
4  const int MAX_SIZE = 20;
5  typedef int infotype;
6
7  struct Stack {
8      infotype info[MAX_SIZE];
9      int top;
10 };
11
12 void createStack(Stack &S);
13 void push(Stack &S, infotype x);
14 infotype pop(Stack &S);
15 void printInfo(const Stack &S);
16 void balikStack(Stack &S);
17
18 #endif
19
```

snappify.com

file stack.cpp

```
1  #include <iostream>
2  #include "stack.h"
3
4  using namespace std;
5
6  void createStack(Stack &S) {
7      S.top = -1; // Initialize stack as empty
8  }
9
10 void push(Stack &S, infotype x) {
11     if (S.top < MAX_SIZE - 1) {
12         S.top++;
13         S.info[S.top] = x;
14     } else {
15         cout << "Stack overflow" << endl;
16     }
17 }
18
19 infotype pop(Stack &S) {
20     if (S.top >= 0) {
21         infotype x = S.info[S.top];
22         S.top--;
23         return x;
24     } else {
25         cout << "Stack underflow" << endl;
26         return -1; // Assuming -1 represents an error or invalid data
27     }
28 }
29
30 void printInfo(const Stack &S) {
31     cout << "[TOP] ";
32     for (int i = S.top; i >= 0; i--) {
33         cout << S.info[i] << " ";
34     }
35     cout << endl;
36 }
37
38 void balikStack(Stack &S) {
39     Stack tempStack;
40     createStack(tempStack);
41
42     while (S.top >= 0) {
43         push(tempStack, pop(S));
44     }
45
46     while (tempStack.top >= 0) {
47         push(S, pop(tempStack));
48     }
49 }
50
```

snappify.com

File main.cpp

```
1  #include <iostream>
2  #include "stack.h"
3
4  using namespace std;
5
6  int main() {
7      cout << "Hello world!" << endl;
8
9      Stack S;
10     createStack(S);
11     push(S, 3);
12     push(S, 4);
13     push(S, 8);
14     pop(S);
15     push(S, 2);
16     push(S, 3);
17     pop(S);
18     push(S, 9);
19
20     printInfo(S);
21
22     cout << "balik stack" << endl;
23     balikStack(S);
24     printInfo(S);
25
26     return 0;
27 }
28
```

snappify.com

Penjelasan Output

- Setelah operasi push dan pop awal, printInfo(S) akan menampilkan isi stack dari atas ke bawah.
- Fungsi balikStack membalik urutan stack, sehingga printInfo(S) terakhir akan menampilkan stack dalam urutan terbalik.

Soal 2

Tambahan di stack.cpp

```
1 void pushAscending(Stack &S, infotype x) {
2     // Buat stack sementara untuk menyimpan elemen yang lebih besar dari x
3     Stack tempStack;
4     createStack(tempStack);
5
6     // Pindahkan elemen yang lebih besar dari x ke tempStack
7     while (S.top ≥ 0 && S.info[S.top] > x) {
8         push(tempStack, pop(S));
9     }
10
11     // Tambahkan x ke stack S
12     push(S, x);
13
14     // Kembalikan elemen dari tempStack ke S
15     while (tempStack.top ≥ 0) {
16         push(S, pop(tempStack));
17     }
18 }
```

snappify.com

Tambahkan deklarasi fungsi pushAscending di file stack.h :

```
1 void pushAscending(Stack &S, infotype x);
2
```

Penjelasan Fungsi pushAscending

1. **Buat Stack Sementara:** Stack sementara (tempStack) dibuat untuk menyimpan elemen-elemen yang lebih besar dari x.
2. **Pindahkan Elemen ke Stack Sementara:** Selama elemen di puncak stack (S) lebih besar dari x, pindahkan elemen tersebut ke tempStack.
3. **Tambahkan Elemen Baru:** Masukkan x ke stack utama S.
4. **Kembalikan Elemen:** Pindahkan kembali elemen-elemen dari tempStack ke stack utama S.

Soal 3

tambahan di stack.cpp

```
1 void getInputStream(Stack &S) {
2     char input;
3     cout << "Masukkan angka-angka secara berurutan, akhiri dengan Enter: ";
4
5     // Menggunakan cin.get() untuk membaca satu karakter setiap kali
6     while (cin.get(input) && input != '\n') {
7         // Memastikan karakter yang dibaca adalah digit
8         if (isdigit(input)) {
9             int num = input - '0'; // Konversi karakter ke integer
10            push(S, num); // Masukkan ke stack
11        }
12    }
13 }
```

tambahan di stack.h

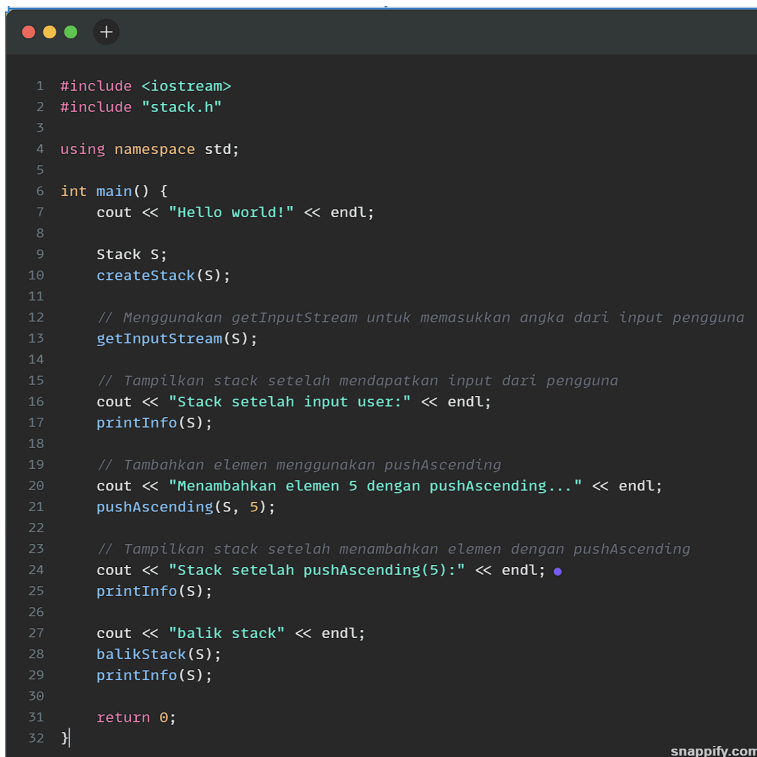
```
1 void getInputStream(Stack &S);
2
3
```

Penjelasan Fungsi getInputStream

1. **Membaca Karakter:** Menggunakan `cin.get(input)` untuk membaca satu karakter pada satu waktu. Proses ini akan terus berjalan hingga pengguna menekan tombol Enter (`'\n'`).
2. **Memeriksa Apakah Karakter Digit:** Setiap karakter yang dimasukkan akan diperiksa apakah itu angka (digit) menggunakan `isdigit(input)`. Jika ya, karakter tersebut akan diubah menjadi integer dengan `input - '0'`.
3. **Push ke Stack:** Integer yang dihasilkan kemudian dimasukkan ke dalam stack menggunakan fungsi `push`.

perubahan di main.cpp

`getInputStream` untuk mendapatkan input awal dari pengguna, dan `pushAscending` untuk menambahkan elemen dalam urutan menaik (penjelasan ada di kodingan)



```
1 #include <iostream>
2 #include "stack.h"
3
4 using namespace std;
5
6 int main() {
7     cout << "Hello world!" << endl;
8
9     Stack S;
10    createStack(S);
11
12    // Menggunakan getInputStream untuk memasukkan angka dari input pengguna
13    getInputStream(S);
14
15    // Tampilkan stack setelah mendapatkan input dari pengguna
16    cout << "Stack setelah input user:" << endl;
17    printInfo(S);
18
19    // Tambahkan elemen menggunakan pushAscending
20    cout << "Menambahkan elemen 5 dengan pushAscending..." << endl;
21    pushAscending(S, 5);
22
23    // Tampilkan stack setelah menambahkan elemen dengan pushAscending
24    cout << "Stack setelah pushAscending(5):" << endl;
25    printInfo(S);
26
27    cout << "balik stack" << endl;
28    balikStack(S);
29    printInfo(S);
30
31    return 0;
32 }
```

Penjelasan Program:

1. **getInputStream(S)**: Mengambil input angka-angka dari pengguna dan memasukkannya ke dalam stack.
2. **printInfo(S)**: Menampilkan isi stack setelah input dari pengguna.
3. **pushAscending(S, 5)**: Menambahkan angka 5 ke stack dengan urutan menaik.
4. **printInfo(S)**: Menampilkan stack setelah `pushAscending` untuk melihat apakah elemen baru telah dimasukkan di posisi yang tepat.
5. **balikStack(S)**: Membalik urutan stack.
6. **printInfo(S)**: Menampilkan stack setelah dibalik.

Soal TP

File stack.cpp

```
1  #include <iostream>
2  #include "stack.h"
3  using namespace std;
4
5  const int MAX_SIZE = 15; // Ukuran maksimum stack
6
7  typedef char infotype;
8
9  struct stack {
10     infotype info[MAX_SIZE];
11     int top;
12 };
13
14 // Membuat stack kosong
15 void createStack(stack &S) {
16     S.top = -1;
17 }
18
19 // Mengecek apakah stack kosong
20 bool isEmpty(stack S) {
21     return S.top == -1;
22 }
23
24 // Mengecek apakah stack penuh
25 bool isFull(stack S) {
26     return S.top == MAX_SIZE - 1;
27 }
28
29 // Menambahkan elemen ke dalam stack
30 void push(stack &S, infotype x) {
31     if (!isFull(S)) {
32         S.top++;
33         S.info[S.top] = x;
34     } else {
35         cout << "Stack penuh, tidak dapat push." << endl;
36     }
37 }
38
39 // Mengeluarkan elemen dari stack
40 infotype pop(stack &S) {
41     if (!isEmpty(S)) {
42         infotype x = S.info[S.top];
43         S.top--;
44         return x;
45     } else {
46         cout << "Stack kosong, tidak dapat pop." << endl;
47         return '\0'; // Mengembalikan karakter kosong jika stack kosong
48     }
49 }
50
51 // Menampilkan semua elemen dalam stack
52 void printInfo(stack S) {
53     if (!isEmpty(S)) {
54         for (int i = 0; i <= S.top; i++) {
55             cout << S.info[i] << " ";
56         }
57         cout << endl;
58     } else {
59         cout << "Stack kosong." << endl;
60     }
61 }
62
```

File main.cpp

```
1 #include "stack.h"
2 #include <iostream>
3 using namespace std;
4
5 // Fungsi untuk menampilkan isi stack secara terbalik (dari top ke bawah)
6 void printReverse(stack S) {
7     while (!isEmpty(S)) {
8         cout << pop(S) << " ";
9     }
10    cout << endl;
11 }
12
13 int main() {
14     stack S;
15
16     // Digit terakhir NIM MOD 4 sisa 0
17     cout << "Digit terakhir NIM MOD 4 sisa 0 : " << endl;
18     createStack(S);
19     push(S, 'I'); push(S, 'F'); push(S, 'L'); push(S, 'A'); push(S, 'B'); push(S, 'J'); push(S, 'A'); push(S, 'Y'); push(S, 'A');
20     cout << "Output: " << endl;
21     printInfo(S); // Isi stack awal
22     cout << endl;
23     pop(S); pop(S); pop(S); pop(S); // Mengeluarkan beberapa elemen
24     printReverse(S); // Isi stack setelah pop
25     cout << endl;
26
27     // Digit terakhir NIM MOD 4 sisa 1
28     cout << "Digit terakhir NIM MOD 4 sisa 1 : " << endl;
29     createStack(S);
30     push(S, 'H'); push(S, 'A'); push(S, 'L'); push(S, 'O'); push(S, 'B'); push(S, 'A'); push(S, 'N'); push(S, 'D'); push(S, 'U'); push(S, 'N'); push(S, 'G');
31     cout << "Output: " << endl;
32     printInfo(S); // Isi stack awal
33     cout << endl;
34     pop(S); pop(S); pop(S); pop(S); pop(S); pop(S); // Mengeluarkan beberapa elemen
35     printReverse(S); // Isi stack setelah pop
36     cout << endl;
37
38     // Digit terakhir NIM MOD 4 sisa 2
39     cout << "Digit terakhir NIM MOD 4 sisa 2 : " << endl;
40     createStack(S);
41     push(S, 'P'); push(S, 'E'); push(S, 'R'); push(S, 'C'); push(S, 'A'); push(S, 'Y'); push(S, 'A'); push(S, 'D'); push(S, 'I'); push(S, 'R'); push(S, 'I');
42     cout << "Output: " << endl;
43     printInfo(S); // Isi stack awal
44     cout << endl;
45     pop(S); pop(S); pop(S); pop(S); pop(S); // Mengeluarkan beberapa elemen
46     printReverse(S); // Isi stack setelah pop
47     cout << endl;
48
49     // Digit terakhir NIM MOD 4 sisa 3
50     cout << "Digit terakhir NIM MOD 4 sisa 3 : " << endl;
51     createStack(S);
52     push(S, 'S'); push(S, 'T'); push(S, 'R'); push(S, 'U'); push(S, 'K'); push(S, 'T'); push(S, 'U'); push(S, 'R'); push(S, 'D'); push(S, 'A'); push(S, 'T'); push(S, 'A');
53     cout << "Output: " << endl;
54     printInfo(S); // Isi stack awal
55     cout << endl;
56     pop(S); pop(S); pop(S); pop(S); pop(S); pop(S); // Mengeluarkan beberapa elemen
57     printReverse(S); // Isi stack setelah pop
58     cout << endl;
59
60     return 0;
61 }
62
```


File stack.h

```
1  #ifndef STACK_H
2  #define STACK_H
3
4  #include <iostream>
5  using namespace std;
6
7  const int MAX_SIZE = 15; // Ukuran maksimum stack
8  typedef char infotype;
9
10 struct stack {
11     infotype info[MAX_SIZE];
12     int top;
13 };
14
15 void createStack(stack &S);
16 bool isEmpty(stack S);
17 bool isFull(stack S);
18 void push(stack &S, infotype x);
19 infotype pop(stack &S);
20 void printInfo(stack S);
21
22 #endif
23
```

snappify.com

Output :

```
noven@NOVEN MINGW64 ~/Desktop/Prak SD/07_Stack/TP
$ ./stack_program
Digit terakhir NIM MOD 4 sisa 0 :
Output:
I F L A B J A Y A

B A L F I

Digit terakhir NIM MOD 4 sisa 1 :
Output:
H A L O B A N D U N G

B O L A H

Digit terakhir NIM MOD 4 sisa 2 :
Output:
P E R C A Y A D I R I

Y A C R E P

Digit terakhir NIM MOD 4 sisa 3 :
Output:
S T R U K T U R D A T A

K U R T S
```

