

# **LAPORAN PRAKTIKUM**

## **PERTEMUAN 7**

### **STACK**



**Nama :**

Alvin Bagus Firmansyah - 2311104070

**Dosen :**

Wahyu Andi Saputra, S.Pd., M.Eng

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## I. TUJUAN

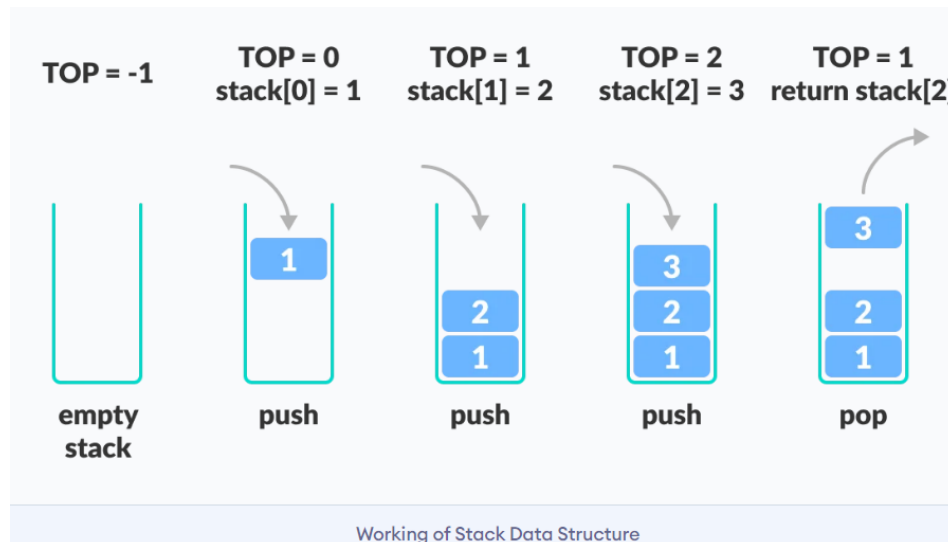
1. Mampu memahami konsep stack pada struktur data dan algoritma
2. Mampu mengimplementasikan operasi-operasi pada stack
3. Mampu memecahkan permasalahan dengan solusi stack

## II. TOOL

IDE/Editor: Program ini dapat dijalankan di IDE seperti Visual Studio Code, Code::Blocks, atau Dev-C++.

## III. DASAR TEORI

Stack adalah struktur data linier yang mengikuti prinsip Last In, First Out (LIFO). Artinya, elemen terakhir yang dimasukkan ke dalam stack akan menjadi elemen pertama yang dikeluarkan.



LIFO: Prinsip dasar stack adalah LIFO. Elemen yang paling baru ditambahkan akan berada di atas dan akan menjadi yang pertama dihapus.

Operasi:

- Push: Menambahkan elemen baru ke atas stack.
- Pop: Menghapus elemen paling atas dari stack.
- Peek: Melihat elemen paling atas tanpa menghapusnya.
- IsEmpty: Mengecek apakah stack kosong.
- IsFull: Mengecek apakah stack penuh (jika stack memiliki kapasitas terbatas).

Representasi:

- Array: Stack dapat diimplementasikan menggunakan array dengan indeks sebagai penunjuk ke elemen teratas.
- Linked List: Stack juga dapat diimplementasikan menggunakan linked list, di mana setiap node menyimpan data dan pointer ke node berikutnya.

## IV. GUIDE

### 1.Codingan

```
main.cpp x main.cpp x main.cpp x include\stackh x stackcpp x main.cpp x main.cpp x
1 #include <iostream>
2 #define MAX 100
3
4 using namespace std;
5
6 class Stack {
7 private:
8     int top;
9     int arr[MAX];
10
11 public:
12     Stack() { top = -1; }
13
14     bool isFull() { return top == MAX - 1; }
15     bool isEmpty() { return top == -1; }
16
17     void push(int x) {
18         if (isFull()) {
19             cout << "Stack Overflow\n";
20             return;
21         }
22         arr[++top] = x;
23     }
24
25     void pop() {
26         if (isEmpty()) {
27             cout << "Stack Underflow\n";
28             return;
29         }
30         top--;
31     }
32
33     int peek() {
34         if (!isEmpty()) {
35             return arr[top];
36         }
37         cout << "Stack is empty\n";
38         return -1;
39     }
40 }
```

```
40
41
42 void display() {
43     if (isEmpty()) {
44         cout << "Stack is empty\n";
45         return;
46     }
47     for (int i = top; i >= 0; i--) {
48         cout << arr[i] << " ";
49     }
50     cout << "\n";
51 }
52
53 int main() {
54     Stack s;
55     s.push(10);
56     s.push(20);
57     s.push(30);
58
59     cout << "Elemen stack: ";
60     s.display();
61
62     cout << "Elemen teratas: " << s.peek() << "\n";
63
64     s.pop();
65     s.pop();
66     cout << "Setelah pop, elemen stack: ";
67     s.display();
68
69     return 0;
70 }
71 }
```

### Hasil Outputnya:

```
"C:\Users\alvin\OneDrive\Dot . x + v
Elemen stack: 30 20 10
Elemen teratas: 30
Setelah pop, elemen stack: 10

Process returned 0 (0x0)   execution time : 1.401 s
Press any key to continue.
```

### 2.Codingan

```
main.cpp X
1  #include <iostream>
2  using namespace std;
3
4  class Node {
5  public:
6      int data;
7      Node* next;
8      Node* Node::next
9  Node(int InputIterator std::next(_InputIterator __x, typename iterator_traits<_InputIterator>::difference_type __n)
10     data = value;
11     next = nullptr;
12 }
13 };
14
15 class Stack {
16 private:
17     Node* top;
18
19 public:
20     Stack() { top = nullptr; }
21
22     bool isEmpty() { return top == nullptr; }
23
24     void push(int x) {
25         Node* newNode = new Node(x);
26         newNode->next = top;
27         top = newNode;
28     }
29
30     void pop() {
31         if (isEmpty()) {
32             cout << "Stack Underflow\n";
33             return;
34         }
35         Node* temp = top;
36         top = top->next;
37         delete temp;
38     }
39 }
```

```
main.cpp X
40
41 int peek() {
42     if (!isEmpty()) {
43         return top->data;
44     }
45     cout << "Stack is empty\n";
46     return -1;
47 }
48
49 void display() {
50     if (isEmpty()) {
51         cout << "Stack is empty\n";
52         return;
53     }
54     Node* current = top;
55     while (current) {
56         cout << current->data << " ";
57         current = current->next;
58     }
59     cout << "\n";
60 }
61
62 int main() {
63     Stack s;
64     s.push(10);
65     s.push(20);
66     s.push(30);
67
68     cout << "Elemen stack: ";
69     s.display();
70
71     cout << "Elemen teratas: " << s.peek() << "\n";
72
73     s.pop();
74     cout << "Setelah pop, elemen stack: ";
75     s.display();
76
77     return 0;
78 }
```

Hasil Outputnya:

```
"C:\Users\alvin\OneDrive\Do... X + v
Elemen stack: 30 20 10
Elemen teratas: 30
Setelah pop, elemen stack: 20 10

Process returned 0 (0x0)   execution time : 10.590 s
Press any key to continue.
```

## V. UNGUIDED

### 1.Program untuk Menentukan Apakah Kalimat Merupakan Palindrom

Codinganya:

```
main.cpp X
1 #include <iostream>
2 #include <cctype> // Untuk fungsi tolower dan isalpha
3 #include <stack> // Untuk stack
4 #include <string>
5 using namespace std;
6
7 bool isPalindrome(const string &str) {
8     stack<char> s;
9
10    // Menyimpan hanya karakter alphanumeric dalam stack
11    for (char c : str) {
12        if (isalnum(c)) {
13            s.push(tolower(c)); // Masukkan karakter ke stack dalam huruf kecil
14        }
15    }
16
17    // Memeriksa karakter dari depan dan belakang
18    int i = 0;
19    while (i < str.size()) {
20        if (isalnum(str[i])) {
21            if (s.top() != tolower(str[i])) {
22                return false; // Jika karakter tidak sama, bukan palindrom
23            }
24            s.pop(); // Hapus elemen dari stack jika cocok
25        }
26        i++;
27    }
28
29    return true; // Jika seluruh karakter cocok
30 }
31
32 int main() {
33     string input;
34
35     cout << "Masukkan kalimat: ";
36     getline(cin, input);
37
38     if (isPalindrome(input)) {
39         cout << "Kalimat tersebut adalah palindrom.\n";
40
41     } else {
42         cout << "Kalimat tersebut bukan palindrom.\n";
43     }
44
45     return 0;
46 }
```

Hasil Outputnya:

```
"C:\Users\alvin\OneDrive\Dot... X + v
Masukkan kalimat: A man a plan a canal Panama
Kalimat tersebut adalah palindrom.

Process returned 0 (0x0)   execution time : 20.250 s
Press any key to continue.
```

## 2. Program untuk Membalikkan Kalimat Menggunakan Stack.

```
main.cpp X
1 #include <iostream>
2 #include <stack>
3 #include <sstream> // Untuk istringstream
4 #include <vector>
5 using namespace std;
6
7 void reverseSentence(string sentence) {
8     stack<string> words;
9     string word;
10
11    // Memecah kalimat menjadi kata-kata dan memasukkan ke stack
12    istringstream iss(sentence);
13    while (iss >> word) {
14        words.push(word);
15    }
16
17    // Mencetak kata-kata dari stack (dalam urutan terbalik)
18    cout << "Kalimat setelah dibalik: ";
19    while (!words.empty()) {
20        cout << words.top() << " ";
21        words.pop();
22    }
23    cout << endl;
24 }
25
26 int main() {
27     string input;
28
29     cout << "Masukkan kalimat (minimal 3 kata): ";
30     getline(cin, input);
31
32     reverseSentence(input); // Memanggil fungsi untuk membalikkan kalimat
33
34     return 0;
35 }
36 }
```

Hasil Outputnya:

A screenshot of a Windows command prompt window. The title bar shows the file path "C:\Users\alvin\OneDrive\Doi". The command prompt displays the following text: "Masukkan kalimat (minimal 3 kata): Hello world from stack", "Kalimat setelah dibalik: stack from world Hello", "Process returned 0 (0x0) execution time : 18.137 s", and "Press any key to continue." The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

## VI. KESIMPULAN

Konsep Stack: Stack adalah struktur data yang mengikuti prinsip Last In, First Out (LIFO).

Elemen terakhir yang dimasukkan adalah yang pertama dikeluarkan.

Implementasi Operasi Stack: Dengan menggunakan array atau linked list, kita dapat mengimplementasikan operasi dasar stack, seperti push, pop, peek, isEmpty, dan isFull.

Palindrom dengan Stack: Program untuk memeriksa apakah kalimat adalah palindrom bekerja dengan membandingkan karakter yang dibaca dari depan dengan karakter yang dibaca dari belakang menggunakan stack.

Pembalikan Kalimat dengan Stack: Pembalikan kalimat menggunakan stack dilakukan dengan memasukkan setiap kata ke dalam stack, dan kemudian mengeluarkannya satu per satu untuk mendapatkan urutan yang terbalik.