

**LAPORAN PRAKTIKUM
STRUKTUR DATA
MODUL 7
“ STACK ”**



Disusun Oleh:
Dhiya Ulhaq Ramadhan 2211104053
Kelas :
S1SE-07-02
Dosen :
Wahyu Andi Saputra, S.Pd., M.Eng.

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024**

1. Tujuan

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

2. Landasan Teori

Stack atau tumpukan merupakan struktur data yang berfungsi untuk menyimpan dan mengelola kumpulan data dengan prinsip Last-In, First-Out (LIFO). Analogi yang sering digunakan adalah tumpukan piring di kafetaria, di mana piring terakhir yang ditambahkan akan menjadi yang pertama diambil. Prinsip ini membuat stack menjadi struktur data yang sangat efisien untuk kasus-kasus tertentu dalam pemrograman.

Dalam implementasinya, stack dapat direpresentasikan sebagai struktur data terurut yang memiliki dua operasi utama yaitu push dan pop. Operasi push digunakan untuk menambahkan elemen baru ke dalam stack, sementara operasi pop digunakan untuk menghapus elemen teratas dari stack. Selain itu, terdapat beberapa operasi pendukung seperti peek untuk melihat elemen teratas tanpa menghapusnya, isEmpty untuk mengecek apakah stack kosong, dan isFull untuk memeriksa apakah stack sudah penuh.

Dengan prinsip LIFO, stack memungkinkan akses data dengan efisiensi tinggi, di mana elemen yang terakhir dimasukkan akan menjadi yang pertama diambil. Karakteristik ini menjadikan stack sebagai salah satu struktur data fundamental yang sangat penting dalam pengembangan perangkat lunak dan pemrograman komputer. Kemampuan stack untuk mengelola data secara terurut dan efisien membuatnya menjadi pilihan ideal untuk berbagai aplikasi yang membutuhkan pengelolaan data dengan pola akses last-in-first-out.

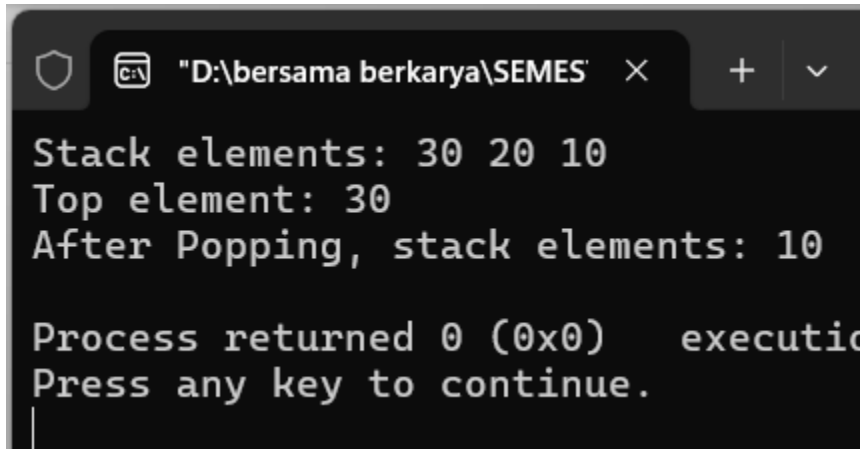
3. Guided 1

Source code :

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

```
34 int peek() {
35     if (!isEmpty()) {
36         return arr[top];
37     }
38     cout << "Stack is Empty\n";
39     return -1;
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 << endl;
51 }
52 };
53
54 int main() {
55     Stack s;
56     s.push(10);
57     s.push(20);
58     s.push(30);
59
60     cout << "Stack elements: ";
61     s.display();
62
63     cout << "Top element: " << s.peek() << "\n";
64
65     s.pop();
66     s.pop();
67     cout << "After Popping, stack elements: ";
68     s.display();
69
70     return 0;
71 }
```

Output :



```
"D:\bersama berkarya\SEMES"
Stack elements: 30 20 10
Top element: 30
After Popping, stack elements: 10
Process returned 0 (0x0) execution completed.
Press any key to continue.
```

Deskripsi program

Program ini mengimplementasikan struktur data Stack menggunakan array dengan kapasitas maksimum 100 elemen. Program terdiri dari sebuah class Stack yang memiliki dua atribut private yaitu top sebagai penanda indeks teratas dan arr sebagai array untuk menyimpan data.

Class Stack memiliki beberapa fungsi utama. Constructor berfungsi untuk menginisialisasi top dengan nilai -1 yang menandakan stack kosong. Fungsi isFull() digunakan untuk memeriksa apakah stack penuh dengan membandingkan top dengan MAX-1. Fungsi isEmpty() memeriksa apakah stack kosong dengan melihat nilai top sama dengan -1. Fungsi push(x) bertugas menambah elemen x ke posisi teratas stack. Fungsi pop() digunakan untuk menghapus elemen teratas dari stack. Fungsi peek() akan mengembalikan nilai elemen teratas tanpa menghapusnya. Sedangkan fungsi display() bertugas menampilkan seluruh elemen stack dari atas ke bawah

4. Guided 2

```
76 //Guided 2
77 #include <iostream>
78 using namespace std;
79
80 class Node {
81 public:
82     int data;
83     Node* next;
84
85     Node(int value) {
86         data = value;
87         next = nullptr;
88     }
89 };
90
91 class Stack {
92 private:
93     Node* top;
94
95 public:
96     Stack() { top = nullptr; }
97
98     bool isEmpty() { return top == nullptr; }
99
100    void push(int x) {
101        Node* newNode = new Node(x);
102        newNode->next = top;
103        top = newNode;
104    }
```

```
106 void pop() {
107     if (isEmpty()) {
108         cout << "Stack Underflow\n";
109         return;
110     }
111     Node* temp = top;
112     top = top->next;
113     delete temp;
114 }
115
116 int peek() {
117     if (isEmpty()) {
118         return top->data;
119     }
120     cout << "Stack is Empty\n";
121     return -1;
122 }
123
124 void display() {
125     if (isEmpty()) {
126         cout << "Stack is Empty\n";
127         return;
128     }
129     Node* current = top;
130     while (current) {
131         cout << current->data << " ";
132         current = current->next;
133     }
134     cout << "\n";
135 }
136 };
137
138 int main() {
139     Stack s;
140
141     s.push(10);
142     s.push(20);
143     s.push(30);
144
145     cout << "Stack elements: ";
146     s.display();
147
148     cout << "Top element: " << s.peek() << "\n";
149
150     s.pop();
151     s.pop();
152
153     cout << "After popping two elements: ";
154     s.display();
155
156     s.pop();
157     cout << "After popping all elements: ";
158     s.display();
159
160     s.pop();
161
162     return 0;
163 }
```

Output :

```
"D:\bersama berkarya\SEMES" x + v
Stack elements: 30 20 10
Top element: Stack is Empty
-1
After popping two elements: 10
After popping all elements: Stack is Empty
Stack Underflow

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

Deskripsi Program

Program terdiri dari dua class yaitu Node untuk membuat node linked list dan Stack untuk implementasi operasi stack. Class Node memiliki atribut data untuk menyimpan nilai dan pointer next untuk menunjuk ke node berikutnya. Sedangkan class Stack memiliki pointer top yang menunjuk ke node teratas. Untuk function masih sama seperti guided 1, jadi langsung saja ke alur main nya

Program dimulai dengan membuat objek Stack s. Kemudian dilakukan operasi push untuk memasukkan tiga elemen yaitu 10, 20, dan 30 secara berurutan ke dalam stack. Program lalu menampilkan seluruh elemen stack menggunakan fungsi display().

Setelah itu program menampilkan elemen teratas menggunakan fungsi peek().

Selanjutnya dilakukan dua kali operasi pop() untuk menghapus dua elemen teratas dan menampilkan sisa elemen. Terakhir dilakukan pop() lagi untuk mengosongkan stack dan mencoba pop() pada stack kosong untuk menguji penanganan error.

Output pertama "Stack elements: 30 20 10" menampilkan isi stack setelah tiga kali push, dimana 30 berada di posisi teratas sesuai prinsip LIFO. Output kedua "Top element: 30" menunjukkan nilai teratas stack yang diperoleh dari fungsi peek().

Setelah dua kali pop yang menghapus 30 dan 20, output ketiga menampilkan sisa satu elemen yaitu 10. Ketika semua elemen dihapus dengan pop terakhir, program menampilkan "Stack is Empty". Percobaan pop pada stack kosong menghasilkan pesan error "Stack Underflow".

1. Buatlah program untuk menentukan apakah kalimat tersebut yang diinputkan dalam program stack adalah palindrom/tidak. Palindrom kalimat yang dibaca dari depan dan belakang sama. Jelaskan bagaimana cara kerja programnya.

Contoh:

Kalimat : ini

Kalimat tersebut adalah polindrom

Kalimat : telkom

Kalimat tersebut adalah bukan polindrom

```
Masukan Kalimat : ini
Kalimat tersebut adalah : Palindrom
```

Jawaban :

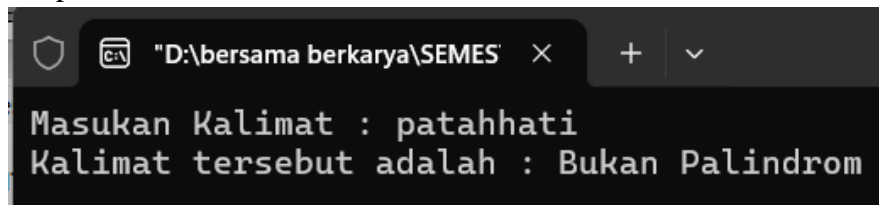
Source code

```
1  #include <iostream>
2  #include <stack>
3  #include <string>
4  using namespace std;
5
6  bool isPalindrom(string kata) {
7      stack<char> s;
8      string cleanStr = "";
9
10     for(char c : kata) {
11         if(c != ' ') {
12             cleanStr += tolower(c);
13         }
14     }
15
16     int length = cleanStr.length();
17     int mid = length / 2;
18
19     for(int i = 0; i < mid; i++) {
20         s.push(cleanStr[i]);
21     }
22
23     int startIndex = (length % 2 == 0) ? mid : mid + 1;
24
25     for(int i = startIndex; i < length; i++) {
26         if(s.empty() || s.top() != cleanStr[i]) {
27             return false;
28         }
29         s.pop();
30     }
```

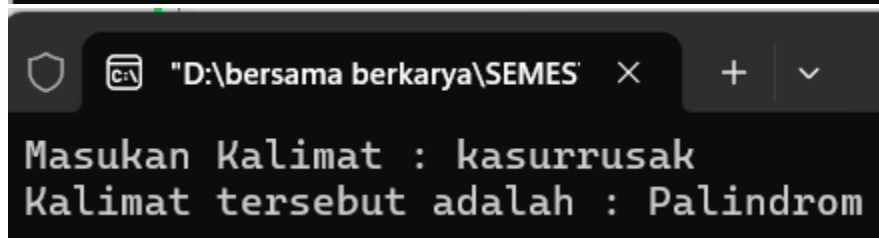


```
32     return s.empty();
33 }
34
35 int main() {
36     string kata;
37     cout << "Masukan Kalimat : ";
38     getline(cin, kata);
39
40     cout << "Kalimat tersebut adalah : ";
41     if(isPalindrom(kata)) {
42         cout << "Palindrom" << endl;
43     } else {
44         cout << "Bukan Palindrom" << endl;
45     }
46
47     return 0;
48 }
```

Output :



```
"D:\bersama berkarya\SEMES" x + v
Masukan Kalimat : patahhati
Kalimat tersebut adalah : Bukan Palindrom
```



```
"D:\bersama berkarya\SEMES" x + v
Masukan Kalimat : kasurrusak
Kalimat tersebut adalah : Palindrom
```

Deskripsi program :

Program ini adalah program untuk mengecek apakah sebuah kalimat termasuk palindrom atau bukan menggunakan bahasa C++. Program menggunakan beberapa library yaitu iostream untuk input/output, stack untuk penggunaan struktur data stack, dan string untuk manipulasi string.

Program terdiri dari dua fungsi utama. Fungsi pertama adalah isPalindrom() yang bertugas melakukan pengecekan palindrom. Fungsi kedua adalah main() yang menangani input dari user dan menampilkan hasil.

Di dalam fungsi isPalindrom(), program pertama-tama membersihkan string input dengan menghapus spasi dan mengubah semua huruf menjadi huruf kecil. Hasil pembersihan disimpan dalam variabel cleanStr. Selanjutnya, program menghitung panjang string dan menentukan titik tengahnya. Proses pengecekan palindrom menggunakan struktur data stack. Setengah pertama dari string yang telah dibersihkan dimasukkan ke dalam stack. Kemudian, setengah kedua string dibandingkan dengan isi

stack. Jika semua karakter cocok dan stack kosong di akhir proses, maka string tersebut adalah palindrom.

Dalam fungsi main(), program meminta user memasukkan kalimat menggunakan `getline(cin, kata)`. Setelah itu, program memanggil fungsi `isPalindrom()` untuk mengecek apakah input tersebut palindrom atau bukan. Hasil pengecekan ditampilkan ke layar. Sebagai contoh, jika user memasukkan "kasur rusak", program akan membersihkan string menjadi "kasur rusak". Karakter "kasur" akan dimasukkan ke stack, kemudian dibandingkan dengan "rusak". Karena setiap karakter cocok dan stack kosong di akhir, program akan menampilkan bahwa kalimat tersebut adalah palindrom.

2. Buatlah program untuk melakukan pembalikan terhadap kalimat menggunakan stack dengan minimal 3 kata. Jelaskan output program dan source codenya beserta operasi/fungsi yang dibuat?

Contoh

Kalimat : Telkom Purwokerto

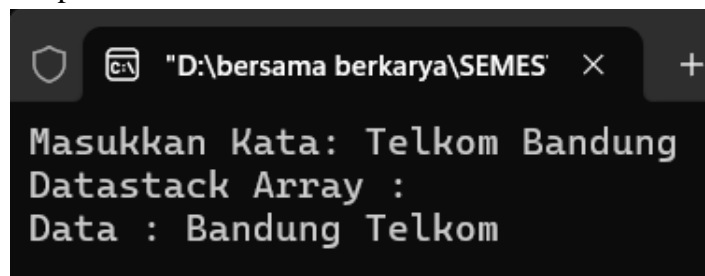
Hasil : otrekowruP mokleT

```
Masukkan Kata Telkom Purwokerto
Datastack Array :
Data : otrekowruP mokleT
```

Source code :

```
53 #include <iostream>
54 #include <stack>
55 #include <string>
56 #include <sstream>
57 using namespace std;
58
59 string balikKalimat(string kalimat) {
60     stack<string> tumpukan;
61     stringstream ss(kalimat);
62     string kata, hasil = "";
63
64     while (ss >> kata) {
65         tumpukan.push(kata);
66     }
67
68     while (!tumpukan.empty()) {
69         hasil += tumpukan.top();
70         tumpukan.pop();
71         if (!tumpukan.empty()) {
72             hasil += " ";
73         }
74     }
75     return hasil;
76 }
77
78 int main() {
79     string kalimat;
80
81     cout << "Masukkan Kata: ";
82     getline(cin, kalimat);
83
84     cout << "Datastack Array : " << endl;
85     cout << "Data : " << balikKalimat(kalimat) << endl;
86
87     return 0;
88 }
```

Output :



```
"D:\bersama berkarya\SEMES" x +
Masukkan Kata: Telkom Bandung
Datastack Array :
Data : Bandung Telkom
```

Deskripsi program

Program ini dibuat untuk melakukan pembalikan kalimat menggunakan struktur data stack dalam bahasa C++. Program menggunakan beberapa library yaitu `iostream` untuk input/output, `stack` untuk penggunaan struktur data stack, `string` untuk manipulasi string, dan `sstream` untuk memisahkan string menjadi kata-kata. Program terdiri dari dua fungsi utama. Fungsi pertama adalah `balikKalimat()` yang bertugas melakukan pembalikan kalimat menggunakan stack. Fungsi kedua adalah `main()` yang menangani input dari user dan menampilkan hasil.

Di dalam fungsi `balikKalimat()`, program menggunakan `stringstream` untuk

memisahkan kalimat input menjadi kata-kata individual. Setiap kata yang dipisahkan kemudian di-push ke dalam stack. Setelah semua kata masuk ke stack, program melakukan pop untuk mengambil kata-kata tersebut dan menyusunnya kembali menjadi kalimat. Karena sifat LIFO (Last In First Out) dari stack, urutan kata-kata akan terbalik.

Dalam fungsi main(), program meminta user memasukkan kalimat menggunakan getline(cin, kalimat). Setelah itu, program memanggil fungsi balikKalimat() untuk membalik urutan kata-kata dalam kalimat tersebut. Hasil pembalikan ditampilkan ke layar.

Sebagai contoh, jika user memasukkan "Telkom Purwokerto", proses yang terjadi adalah:

- Kata "Telkom" dan "Bandung" dipisahkan
- "Telkom" di-push ke stack
- "Bandung" di-push ke stack
- Stack melakukan pop "Bandung"
- Stack melakukan pop "Telkom"
- Hasil digabungkan menjadi "Bandung Telkom"

Output program akan menampilkan:

Masukkan Kata: Telkom Purwokerto

Datastack Array :

Data : Purwokerto Telkom

Kesimpulan

Melalui dua program yang telah dipelajari, yaitu program palindrom dan program pembalikan kalimat, saya dapat melihat bagaimana stack dapat diaplikasikan untuk menyelesaikan masalah yang berbeda. Program palindrom menggunakan stack untuk menyimpan dan membandingkan karakter, sementara program pembalikan kalimat menggunakan stack untuk mengubah urutan kata.