**MODUL 7**

**STACK**

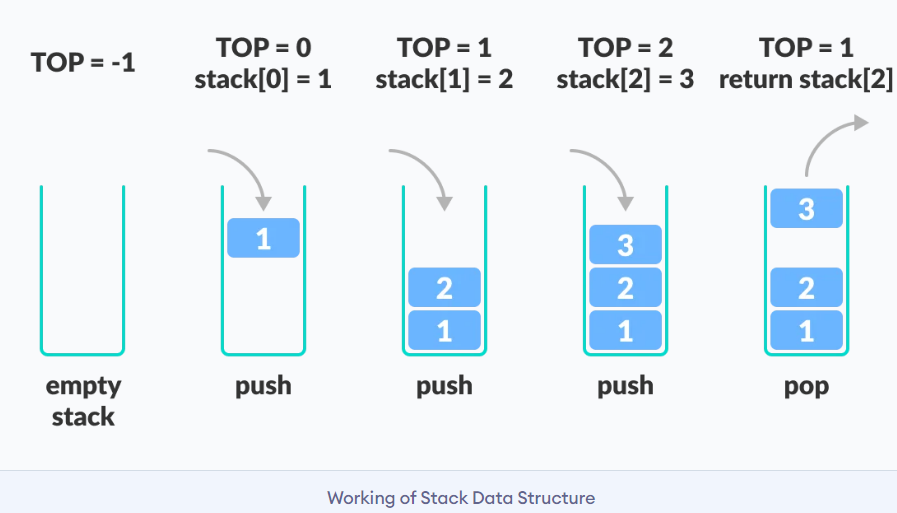
1. **TUJUAN PRAKTIKUM**
2. Mampu memahami konsep stack pada struktur data dan algoritma
3. Mampu mengimplementasikan operasi-operasi pada stack
4. Mampu memecahkan permasalahan dengan solusi stack
5. **DASAR TEORI**

Sebuah 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.

Dalam implementasinya, stack dapat direpresentasikan sebagai struktur data terurut yang memiliki dua operasi utama: push dan pop. Operasi push digunakan untuk menambahkan elemen baru ke dalam stack, sementara operasi pop digunakan untuk menghapus elemen teratas dari stack.

Prinsip LIFO yang menjadi dasar stack membuatnya sangat bermanfaat dalam berbagai aplikasi, termasuk manajemen memori komputer, evaluasi ekspresi aritmatika, dan manajemen panggilan fungsi dalam pemrograman. Sebagai contoh, dalam manajemen memori, stack digunakan untuk menyimpan alamat-alamat memori yang dialokasikan untuk variabel dan fungsi.

Dengan prinsip LIFO ini, stack memungkinkan akses data dengan efisiensi, di mana elemen yang terakhir dimasukkan akan menjadi yang pertama diambil. Hal ini menjadikannya salah satu struktur data yang sangat penting dalam pengembangan perangkat lunak dan pemrograman komputer..



Operasi pada stack melibatkan beberapa fungsi dasar yang dapat dilakukan pada struktur data ini. Berikut adalah beberapa operasi umum pada stack:

1. Push (Masukkan): Menambahkan elemen ke dalam tumpukan pada posisi paling atas atau ujung.
2. Pop (Keluarkan): Menghapus elemen dari posisi paling atas atau ujung tumpukan.
3. Top (Atas): Mendapatkan nilai atau melihat elemen teratas pada tumpukan tanpa menghapusnya.
4. IsEmpty (Kosong): Memeriksa apakah tumpukan kosong atau tidak.
5. IsFull (Penuh): Memeriksa apakah tumpukan penuh atau tidak (terutama pada implementasi tumpukan dengan kapasitas terbatas).
6. Size (Ukuran): Mengembalikan jumlah elemen yang ada dalam tumpukan.
7. Peek (Lihat): Melihat nilai atau elemen pada posisi tertentu dalam tumpukan tanpa menghapusnya.
8. Clear (Hapus Semua): Mengosongkan atau menghapus semua elemen dari tumpukan.
9. Search (Cari): Mencari keberadaan elemen tertentu dalam tumpukan.
10. **GUIDED**

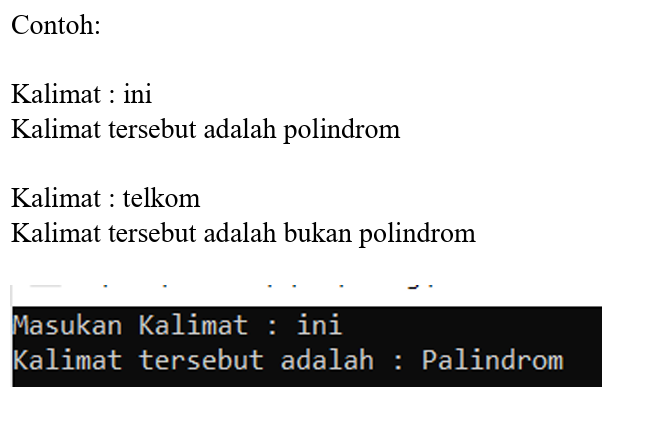
**Sourcecode**

|  |
| --- |
| #include <iostream>  using namespace std;  string arrayBuku[5];  int maksimal = 5, top = 0;  bool isFull() {  return (top == maksimal);  }  bool isEmpty() {  return (top == 0);  }  void pushArrayBuku(string data) {  if (isFull()) {  cout << "Data telah penuh" << endl;  } else {  arrayBuku[top] = data;  top++;  }  }  void popArrayBuku() {  if (isEmpty()) {  cout << "Tidak ada data yang dihapus" << endl;  } else {  arrayBuku[top - 1] = "";  top--;  }  }  void peekArrayBuku(int posisi) {  if (isEmpty()) {  cout << "Tidak ada data yang bisa dilihat" << endl;  } else {  int index = top;  for (int i = 1; i <= posisi; i++) {  index--;  }  cout << "Posisi ke " << posisi << " adalah " << arrayBuku[index] << endl;  }  }  int countStack() {  return top;  }  void changeArrayBuku(int posisi, string data) {  if (posisi > top) {  cout << "Posisi melebihi data yang ada" << endl;  } else {  int index = top;  for (int i = 1; i <= posisi; i++) {  index--;  }  arrayBuku[index] = data;  }  }  void destroyArraybuku() {  for (int i = top; i >= 0; i--) {  arrayBuku[i] = "";  }  top = 0;  }  void cetakArrayBuku() {  if (isEmpty()) {  cout << "Tidak ada data yang dicetak" << endl;  } else {  for (int i = top - 1; i >= 0; i--) {  cout << arrayBuku[i] << endl;  }  }  }  int main() {  pushArrayBuku("Kalkulus");  pushArrayBuku("Struktur Data");  pushArrayBuku("Matematika Diskrit");  pushArrayBuku("Dasar Multimedia");  pushArrayBuku("Inggris");  cetakArrayBuku();  cout << "\n";  cout << "Apakah data stack penuh? " << isFull() << endl;  cout << "Apakah data stack kosong? " << isEmpty() << endl;  peekArrayBuku(2);  popArrayBuku();  cout << "Banyaknya data = " << countStack() << endl;  changeArrayBuku(2, "Bahasa Jerman");  cetakArrayBuku();  cout << "\n";  destroyArraybuku();  cout << "Jumlah data setelah dihapus: " << top << endl;  cetakArrayBuku();  return 0;  } |

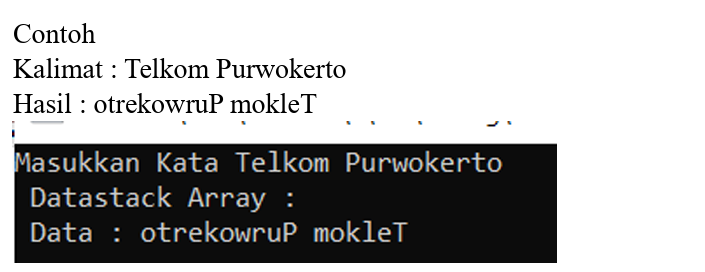
**Output**

|  |
| --- |
| Inggris  Dasar Multimedia  Matematika Diskrit  Struktur Data  Kalkulus  Apakah data stack penuh? 1  Apakah data stack kosong? 0  Posisi ke 2 adalah Dasar Multimedia  Banyaknya data = 4  Dasar Multimedia  Bahasa Jerman  Struktur Data  Kalkulus  Jumlah data setelah dihapus: 0  Tidak ada data yang dicetak |

1. **UNGUIDED**
2. 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.



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



1. **DAFTAR PUSTAKA**

Karumanchi, N. (2016). *Data Structures and algorithms made easy: Concepts, problems, Interview Questions*. CareerMonk Publications.