

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL 6
“STACK”**



**DISUSUN OLEH :
BINTANG YUDHISTIRA
2311102052**

**DOSEN
WAHYU ANDI SAPUTRA, S.PD., M.PD.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

A. Dasar Teori

1. Stack

Stack adalah struktur data linier yang mengikuti aturan tertentu untuk melakukan operasi. Data yang memiliki struktur stack, tersusun seperti tumpukan, sehingga hanya elemen yang baru dimasukkan yang dapat diakses atau dilihat. Ujung tumpukan yang digunakan untuk melakukan semua operasi disebut bagian atas tumpukan. Stack mengikuti prinsip LIFO (Last In First Out), yang berarti elemen yang dimasukkan terakhir akan menjadi elemen pertama yang keluar dari urutan data.

Ujung tumpukan yang digunakan untuk melakukan semua operasi disebut bagian atas tumpukan. Stack mengikuti prinsip LIFO (Last In First Out), yang berarti elemen yang dimasukkan terakhir akan menjadi elemen pertama yang keluar dari urutan data.

2. Queue

Queue adalah struktur data linier di mana kita dapat menyisipkan dan menghapus elemen dari daftar data. Akhir daftar dari mana elemen disisipkan disebut ujung belakang dan ujung di mana elemen dihapus adalah ujung depan.

Struktur data yang menggunakan queue mengikuti prinsip FIFO (First In First Out), yang berarti elemen yang dimasukkan pertama kali dari ujung belakang akan menjadi elemen pertama yang dihapus dari ujung depan. Selain itu, terdapat dua istilah lain dalam queue, yakni operasi enqueue dan operasi dequeue. Operasi enqueue adalah teknik penyisipan pada struktur data queue, sedangkan operasi dequeue adalah teknik penghapusan pada struktur data queue.

Perbedaan Stack dan Queue

Terdapat beberapa perbedaan antara stack dan queue. Berikut beberapa perbedaannya.

- **Struktur Data**

Queue adalah struktur data sederhana yang berfungsi sebagai antrean. Elemen yang ditempatkan pertama kali akan menjadi yang pertama kali diambil, mirip dengan antrean di toko atau bank. Sedangkan Stack adalah struktur data yang memungkinkan penambahan dan penghapusan elemen selalu di akhir atau ujung bawah dan atas, mirip dengan tumpukan buku yang diletakkan satu per satu.

- **Urutan**

Pada Queue, urutan elemen harus dijaga agar elemen yang ditambahkan selalu ditempatkan di ujung belakang, sementara urutan elemen yang diambil dari depan. Namun pada Stack, urutan elemen kurang begitu penting, karena elemen yang ditambahkan terakhir akan selalu diambil terlebih dahulu.

- **Penggunaan**

Queue digunakan pada aplikasi-antrian seperti algoritma BFS (Breadth First Search), dan job queue untuk mengatur antrian pekerjaan. Sedangkan penggunaan Stack lebih memfokuskan kepada pemanggilan fungsi atau prosedur dalam aplikasi, seperti Undo dan Redo di aplikasi pengolah teks.

B. Guided

Guided 1

Source Code :

```
#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;
}

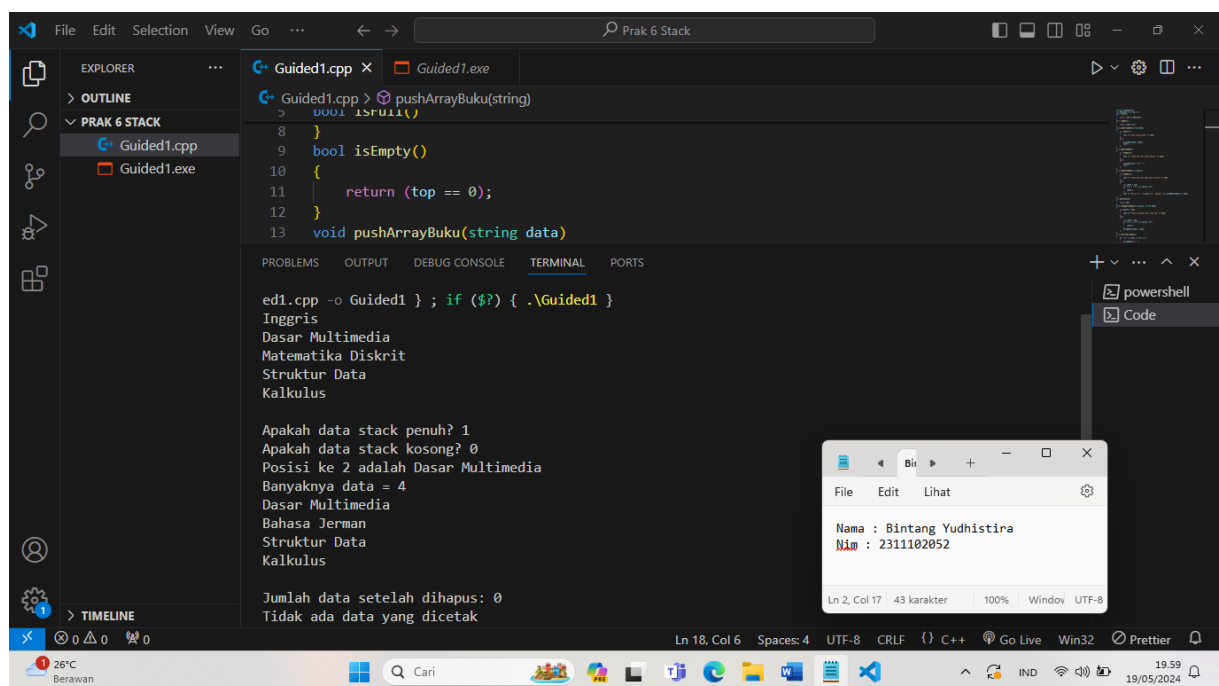
void cetakArrayBuku()
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang dicetak" << endl;
    }
    else
    {
        for (int i = top - 1; i >= 0; i--)
        {
            cout << arrayBuku[i] << endl;
        }
    }
}

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;
}
```

Screenshots Output:



Deskripsi Program :

Kode di atas adalah implementasi dari struktur data stack menggunakan array di bahasa C++. Variabel string arrayBuku[5] digunakan untuk menyimpan data buku dalam array berukuran 5, sedangkan int maksimal = 5, top = 0 menyimpan kapasitas maksimum stack dan indeks teratas stack.

Fungsi `isFull()` mengembalikan `true` jika stack penuh (`top == maksimal`), sedangkan `isEmpty()` mengembalikan `true` jika stack kosong (`top == 0`). Fungsi `pushArrayBuku(string data)` menambahkan data ke stack, dan jika stack penuh, akan menampilkan pesan "Data telah penuh". Fungsi `popArrayBuku()` menghapus data teratas dari stack, dan jika stack kosong, akan menampilkan pesan "Tidak ada data yang dihapus".

Fungsi `peekArrayBuku(int posisi)` menampilkan data pada posisi tertentu dari puncak stack. Jika stack kosong, akan menampilkan pesan "Tidak ada data yang bisa dilihat". Fungsi `countStack()` mengembalikan jumlah data dalam stack (nilai `top`), sementara fungsi `changeArrayBuku(int posisi, string data)` mengubah data pada posisi tertentu dari puncak stack. Jika posisi melebihi jumlah data, akan menampilkan pesan "Posisi melebihi data yang ada".

Fungsi `destroyArraybuku()` menghapus semua data dalam stack dengan mengosongkan array dan mengatur `top` ke 0, sedangkan fungsi `cetakArrayBuku()` mencetak semua data dalam stack dari atas ke bawah. Jika stack kosong, akan menampilkan pesan "Tidak ada data yang dicetak".

Pada fungsi `main()`, pertama-tama lima buku ditambahkan ke stack dan isi stack dicetak. Kemudian, dicek apakah stack penuh atau kosong, buku di posisi kedua dari atas ditampilkan, data teratas dihapus, jumlah data dalam stack ditampilkan, buku di posisi kedua dari atas diubah, dan isi stack kembali dicetak. Terakhir, semua data dalam stack dihapus, jumlah data setelah penghapusan ditampilkan, dan isi stack dicetak lagi. Kode ini memberikan gambaran dasar tentang operasi stack seperti `push`, `pop`, `peek`, `change`, `destroy`, dan `cetak` menggunakan array.

C. Unguided

1. Source Code:

```
#include <iostream>
#include <stack>
#include <cctype>
#include <string>
using namespace std;
bool isPalindrome(const string &str)
{
    stack<char> s;
    string cleanedStr;
    for (char ch : str)
    {
        if (isalpha(ch))
        {
            cleanedStr += tolower(ch);
        }
    }

    for (char ch : cleanedStr)
    {
        s.push(ch);
    }

    for (char ch : cleanedStr)
    {

```

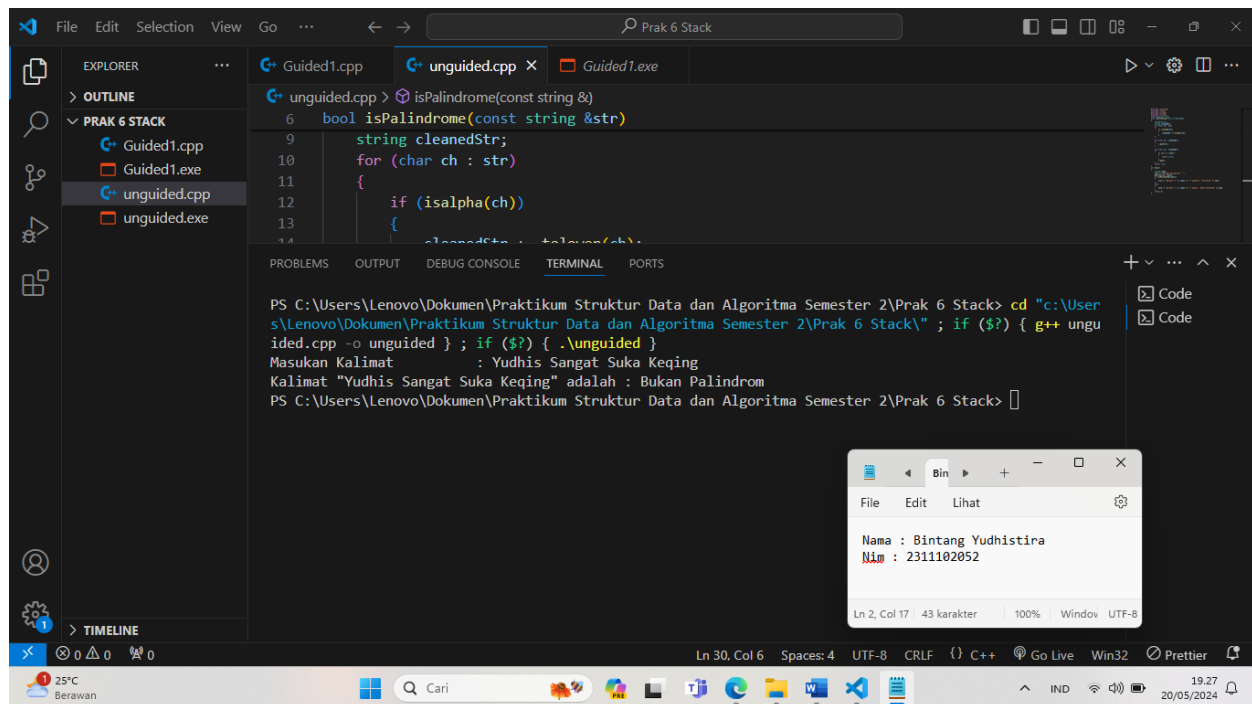


```

        if (ch != s.top())
        {
            return false;
        }
        s.pop();
    }
    return true;
}
int main()
{
    string input;
    cout << "Masukan Kalimat\t\t : ";
    getline(cin, input);
    if (isPalindrome(input))
    {
        cout << "Kalimat \"" << input << "\" adalah\t : Palindrom" <<
endl;
    }
    else
    {
        cout << "Kalimat \"" << input << "\" adalah : Bukan Palindrom"
<< endl;
    }
    return 0;
}

```

Screenshot output



```
6 bool isPalindrome(const string &str)
9     string cleanedStr;
10    for (char ch : str)
11    {
12        if (isalpha(ch))
13        {
14            cleanedStr += tolower(ch);
15        }
16    }
17    stack<char> s;
18    for (char ch : cleanedStr)
19    {
20        s.push(ch);
21    }
22    for (char ch : cleanedStr)
23    {
24        if (ch != s.top())
25        {
26            return false;
27        }
28        s.pop();
29    }
30    return true;
31 }
```

```
PS C:\Users\Lenovo\Dokumen\Praktikum Struktur Data dan Algoritma Semester 2\Prak 6 Stack> cd "c:\Users\Lenovo\Dokumen\Praktikum Struktur Data dan Algoritma Semester 2\Prak 6 Stack" ; if ($?) { g++ unguided.cpp -o unguided } ; if ($?) { .\unguided }
Masukan Kalimat      : Yudhis Sangat Suka Keqing
Kalimat "Yudhis Sangat Suka Keqing" adalah : Bukan Palindrom
PS C:\Users\Lenovo\Dokumen\Praktikum Struktur Data dan Algoritma Semester 2\Prak 6 Stack>
```

```
File Edit Lihat
Nama : Bintang Yudhistira
Nim : 2311102052
Ln 2, Col 17 43 karakter 100% Window UTF-8
```

Deskripsi Program :

Kode di atas adalah program C++ untuk memeriksa apakah sebuah kalimat adalah palindrom. Pertama, program mengimpor pustaka yang diperlukan seperti `iostream`, `stack`, `cctype`, dan `string`. Fungsi `isPalindrome` menerima string sebagai parameter dan menggunakan dua tahap untuk membersihkan dan memeriksa string tersebut.

Pada tahap pertama, karakter dalam string diperiksa satu per satu; jika karakter adalah huruf alfabet, ia dikonversi ke huruf kecil dan ditambahkan ke string `cleanedStr`. Tahap kedua, karakter dalam `cleanedStr` didorong ke `stack`. Kemudian, program membandingkan karakter satu per satu dari `cleanedStr` dengan karakter yang diambil dari `stack` (karakter paling atas). Jika semua karakter cocok, fungsi mengembalikan nilai `true`, yang berarti string tersebut adalah palindrom. Jika ada karakter yang tidak cocok, fungsi mengembalikan nilai `false`.

Di fungsi main, program meminta input dari pengguna dan memeriksa apakah input tersebut adalah palindrom menggunakan fungsi `isPalindrome`. Hasil pemeriksaan kemudian ditampilkan kepada pengguna, menunjukkan apakah kalimat tersebut adalah palindrom atau bukan.

2. Source Kode

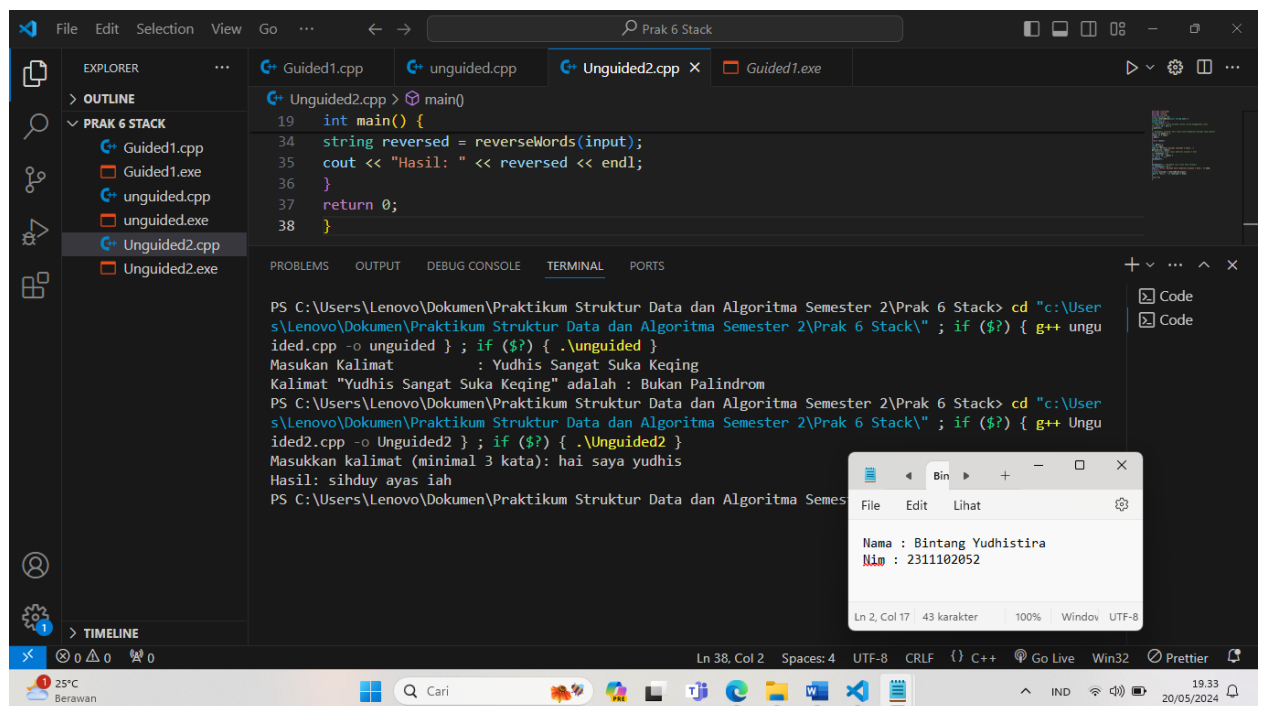
```
#include <iostream>
#include <stack>
#include <string>
using namespace std;
string reverseWords(const string &str)
{
    stack<char> s;
    string result = "";
    // Membalikkan setiap karakter dalam string menggunakan stack
    for (char ch : str)
    {
        s.push(ch);
    }
    // Mengambil karakter dari stack untuk membentuk kalimat yang
    // dibalik
    while (!s.empty())
    {
        result += s.top();
        s.pop();
    }
    return result;
}
int main()
{
    string input;
    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, input);
    // Memastikan bahwa input memiliki minimal 3 kata
    int wordCount = 0;
    for (char ch : input)
    {
        if (ch == ' ')
        {
            wordCount++;
        }
    }
    wordCount++; // Menambah satu untuk kata terakhir
    if (wordCount < 3)
    {
```

```

        cout << "Error: Kalimat harus memiliki minimal 3 kata." << endl;
    }
    else
    {
        string reversed = reverseWords(input);
        cout << "Hasil: " << reversed << endl;
    }
    return 0;
}

```

Screenshoot Output



Deskripsi Program

Kode di atas adalah program C++ yang membalikkan setiap karakter dalam sebuah string yang dimasukkan oleh pengguna, dengan memastikan bahwa input memiliki minimal tiga kata. Program ini menggunakan pustaka `iostream`, `stack`, dan `string`. Fungsi `reverseWords` menerima string sebagai parameter dan menggunakan stack untuk membalikkan urutan karakter dalam string. Setiap karakter dari string dimasukkan ke dalam stack, dan kemudian diambil kembali dari stack untuk membentuk string yang dibalik. Di dalam fungsi `main`, program meminta pengguna untuk memasukkan sebuah kalimat dengan minimal tiga kata. Kemudian, program menghitung jumlah kata dalam kalimat dengan

menghitung jumlah spasi. Jika jumlah kata kurang dari tiga, program menampilkan pesan kesalahan. Jika tidak, program memanggil fungsi `reverseWords` untuk membalikkan karakter dalam kalimat dan menampilkan hasilnya kepada pengguna. Berikut adalah penjelasan langkah demi langkah kode: pertama, program mengimpor pustaka `iostream`, `stack`, dan `string`. Kemudian, fungsi `reverseWords` mendeklarasikan `stack` karakter `s` dan `string` `result`, menggunakan loop untuk menambahkan setiap karakter dari `str` ke dalam `stack`, dan menggunakan loop untuk mengambil setiap karakter dari `stack` dan menambahkannya ke `result`, sehingga menghasilkan `string` yang dibalik. Dalam fungsi `main`, program meminta pengguna untuk memasukkan sebuah kalimat, menghitung jumlah kata dalam kalimat dengan menghitung jumlah spasi, memastikan bahwa kalimat memiliki minimal tiga kata, dan jika jumlah kata kurang dari tiga, menampilkan pesan kesalahan. Jika jumlah kata cukup, program memanggil fungsi `reverseWords` untuk membalikkan kalimat dan menampilkan hasilnya. Contoh output: "Masukkan kalimat (minimal 3 kata): Saya suka koding" akan menghasilkan "Hasil: gnidok akus ayaS".

D. Kesimpulan

Kesimpulan dari materi tentang `stack` adalah bahwa `stack` adalah struktur data yang mengikuti prinsip LIFO (Last In, First Out), di mana elemen terakhir yang dimasukkan adalah elemen pertama yang akan diambil. `Stack` menyediakan operasi dasar seperti `push` (menambahkan elemen ke atas `stack`), `pop` (menghapus elemen dari atas `stack`), dan `top` (mengakses elemen teratas tanpa menghapusnya).

Penggunaan `stack` sangat berguna dalam berbagai aplikasi seperti pemrosesan ekspresi aritmatika, penjagaan status eksekusi dalam pemrograman rekursif, dan algoritma penelusuran graf seperti DFS (Depth First Search). `Stack` juga sering digunakan untuk membalikkan data, seperti dalam contoh program di atas, di mana `stack` digunakan untuk membalikkan karakter dalam `string`.

Secara keseluruhan, `stack` adalah alat yang efektif untuk manajemen data dengan urutan

tertentu, terutama ketika dibutuhkan akses hanya pada elemen terbaru.

E. Referensi

[1] Asisten Pratikum “Modul 6 Stack”, Learning Management System, 2024.

[2] Kompas. Pengertian Stack dan Queue. 1 Desember 2022.

Diakses Pada 19 Mei 2024, dari

[Pengertian Stack dan Queue serta Contoh Penerapannya \(kompas.com\)](https://kompas.com)

[3] Localstartupfest (2023, juli). “Perbedaan Queue dan Stack”.

Diakses pada 19 Mei 2024, dari

[Perbedaan Queue dan Stack: Penjelasan Lengkap Dalam Ilmu Komputer - Localstartupfest.id](https://localstartupfest.id)