LAPORAN PRAKTIKUM STRUKTUR DATA DAN ALGORITMA

MODUL 6

STACK

DISUSUN OLEH:

PRIESTY AMEILIANA MAULIDAH

2311102175

S1 IF-11-E

DOSEN:

Muhammad Afrizal Amrustian, S. Kom.

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO PURWOKERTO

2024

A.DASAR TEORI

Stack adalah struktur data sederhana yang digunakan untuk menyimpan data . Dalam tumpukan, urutan kedatangan data penting. Sebuah tumpukan piring di kafetaria adalah contoh bagus dari tumpukan. Piring ditambahkan ke tumpukan saat mereka dibersihkan dan ditempatkan di bagian atas.

Ketika sebuah piring dibutuhkan, diambil dari bagian atas tumpukan. Piring pertama yang ditempatkan di tumpukan adalah yang terakhir digunakan. Elemen terakhir yang dimasukkan adalah yang pertama dihapus. Operasi pada stack melibatkan beberapa fungsi dasar yang dapat dilakukan pada struktur data ini.

B.Guided

Guided 1

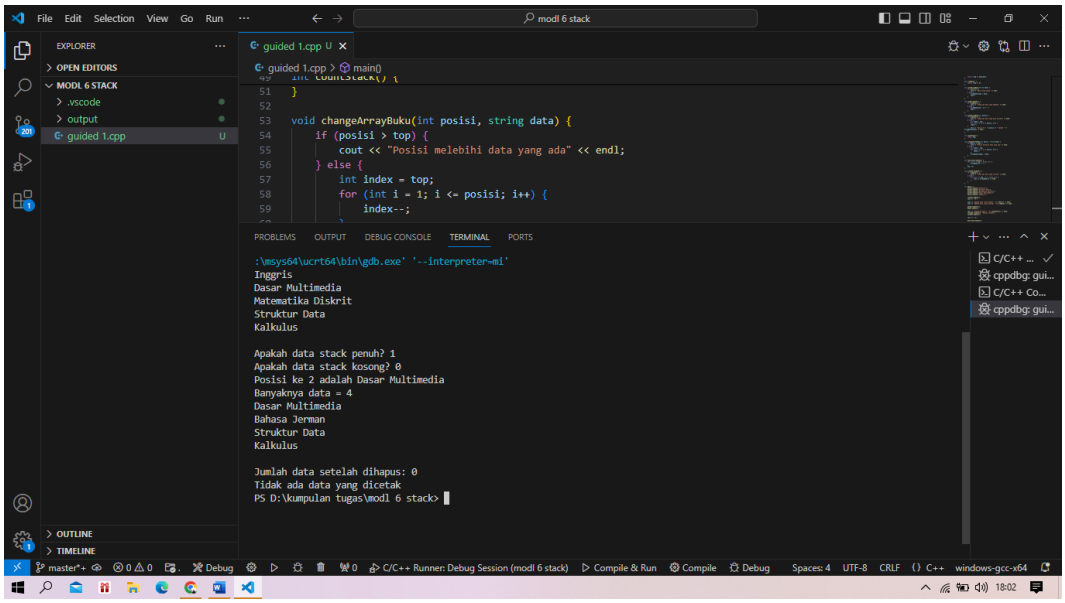
```
// priesty ameiliana maulidah
// 2311102175
#include <iostream>
using namespace std;
string arrayBuku[5];
int maksimal = 5, top = 0;
bool isFull() {
return (top == maksimal);
}
bool isEmpty() {
return (top == 0);
}
void pushArrayBuku(string data) {
if (isFull()) {
cout << "Data telah penuh" << endl;
} 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:

1. **isFull()**: Fungsi ini memeriksa apakah tumpukan sudah penuh atau belum. Mengembalikan **true** jika tumpukan penuh dan **false** jika tidak.
2. **isEmpty()**: Fungsi ini memeriksa apakah tumpukan kosong atau tidak. Mengembalikan **true** jika tumpukan kosong dan **false** jika tidak.
3. **pushArrayBuku(string data)**: Fungsi ini menambahkan data baru ke tumpukan. Jika tumpukan penuh, pesan "Data telah penuh" akan dicetak.
4. **popArrayBuku()**: Fungsi ini menghapus data teratas dari tumpukan. Jika tumpukan kosong,

pesan "Tidak ada data yang dihapus" akan dicetak.

5. **peekArrayBuku(int posisi)**: Fungsi ini menampilkan data pada posisi tertentu dalam tumpukan. Jika tumpukan kosong, pesan "Tidak ada data yang bisa dilihat" akan dicetak.
6. **countStack()**: Fungsi ini mengembalikan jumlah data dalam tumpukan.
7. **changeArrayBuku(int posisi, string data)**: Fungsi ini mengubah data pada posisi tertentu dalam tumpukan dengan data baru.
8. **destroyArraybuku()**: Fungsi ini menghapus semua data dalam tumpukan dan mengatur ulang indeks tumpukan.
9. **cetakArrayBuku()**: Fungsi ini mencetak semua data dalam tumpukan, dimulai dari data teratas.

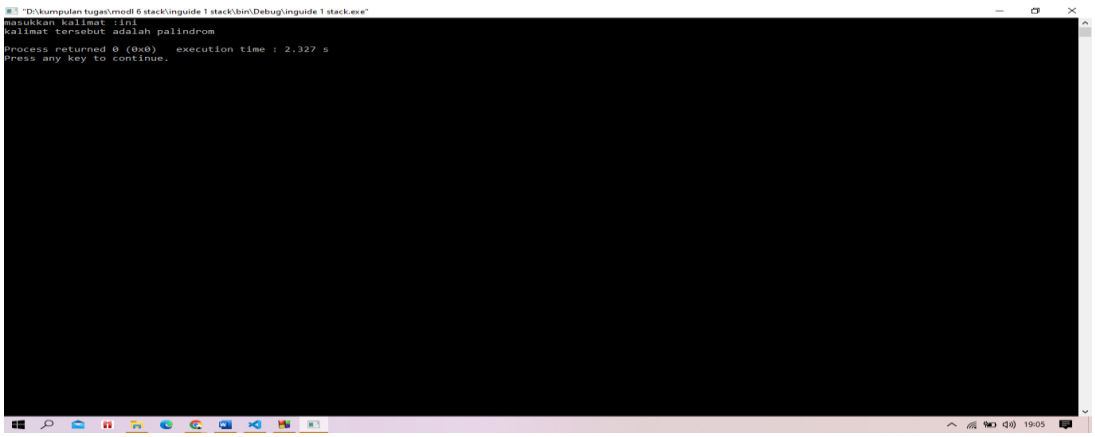
c. unguided/tugas

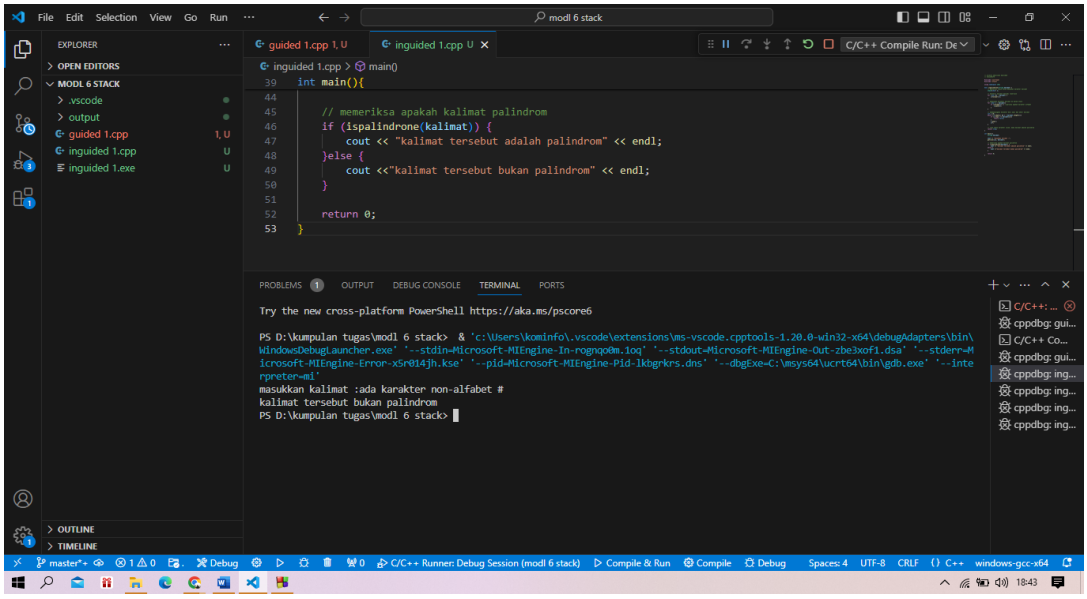
unguided 1

```
// priesty ameiliana maulidah
// 2311102175
#include <iostream>
#include <stack>

using namespace std;

bool ispalindrone(string kalimat) {
   // deklarasi stack untuk menyikan karakter kalimat
   stack<char> s;

   // konversi kalimat menjadi lowercase
   for (char &c : kalimat) {
      c=tolower(c);
   }
// memasukan karakter kalimat ke dalam stack
   for (char c : kalimat){
      if (isalpha(c)) { //periksa apakah karakter alfabet
         s.push(c);
      }
   }
```

```
// membandingkan karakter dari awal dan akhir kalimat
   int i =0;
   while (!s.empty() && i < kalimat.length()){
      if (s.top() != kalimat[i])
         return false;
      }
      s.pop();
      i++;
   }

   // jika semua karakter cocok, maka kalimat adalah palindrom
   return true;
}
int main(){
   string kalimat;

   cout << "masukkan kalimat :";
   getline(cin, kalimat);

   // memeriksa apakah kalimat palindrom
   if (ispalindrone(kalimat)) {
      cout << "kalimat tersebut adalah palindrom" << endl;
   }else {
      cout <<"kalimat tersebut bukan palindrom" << endl;
   }

   return 0;
}
```

Screenshots output





Deskripsi :

- **ispalindrome(string kalimat)**: Fungsi ini memeriksa apakah kalimat yang diberikan merupakan palindrom.
- **Deklarasi stack**: Sebuah stack s dideklarasikan untuk menyimpan karakter-karakter kalimat.

- **Konversi kalimat ke lowercase**: Setiap karakter dalam kalimat diubah menjadi huruf kecil menggunakan tolower().
- **Memasukkan karakter kalimat ke stack**: Setiap karakter alfabet dalam kalimat dimasukkan ke stack s menggunakan push().
- **Membandingkan karakter dari awal dan akhir**: Program melakukan iterasi melalui kalimat dan stack secara bersamaan. Pada setiap iterasi:
    - Karakter teratas stack (s.top()) dibandingkan dengan karakter pada indeks i dalam kalimat.
    - Jika karakter tidak sama, maka kalimat bukan palindrom dan fungsi mengembalikan false.
    - Jika karakter sama, karakter teratas stack dihapus (s.pop()) dan nilai i ditingkatkan.
- **Memeriksa palindrom**: Jika iterasi selesai dan semua karakter cocok, maka kalimat adalah palindrom dan fungsi mengembalikan true.
- **main()**: Fungsi main() adalah fungsi utama program.
- **Meminta input kalimat**: Pengguna diminta untuk memasukkan kalimat.

- **Memeriksa palindrom**: Fungsi ispalindrome dipanggil untuk memeriksa apakah kalimat yang dimasukkan merupakan palindrom.
- **Menampilkan hasil**: Hasil pemeriksaan palindrom dicetak ke konsol.

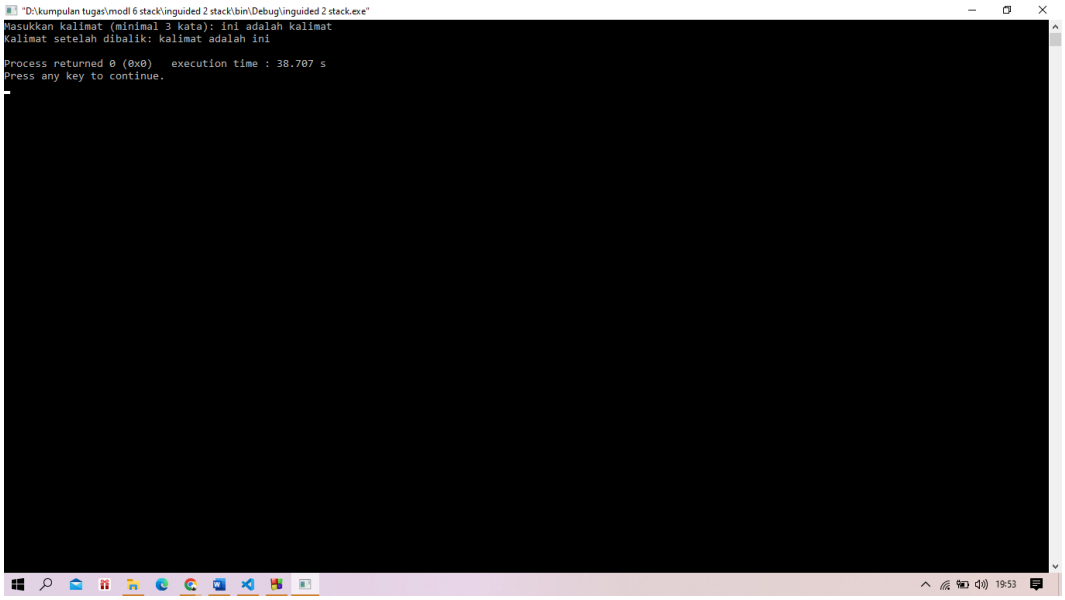
unguided 2

```
  // priesty ameiliana mualidah
// 2311102175


#include <iostream>
#include <stack>
#include <string>


using namespace std;


// Fungsi untuk melakukan pembalikan kalimat menggunakan stack
string reverseSentence(string kalimat) {
    stack<string> kataStack; // Membuat stack untuk menyimpan kata-kata


// Variabel sementara untuk menyimpan setiap kata dalam kalimat
    string kataSementara = "";


    // Memisahkan kalimat menjadi kata-kata
    for (char c : kalimat) {
        if (c == ' ') {
```

```
// Jika menemukan spasi, maka kata sementara disimpan ke dalam stack
            kataStack.push(kataSementara);
            kataSementara = ""; // Mengosongkan kata sementara untuk menampung kata berikutnya
        } else {
            // Jika belum menemukan spasi, maka karakter ditambahkan ke kata sementara
            kataSementara += c;
        }
    }
// Menambahkan kata terakhir setelah loop berakhir
    kataStack.push(kataSementara);

    // Membuat kalimat baru dengan mengambil kata-kata dari stack
    string kalimatTerbalik = "";
    while (!kataStack.empty()) {
        kalimatTerbalik += kataStack.top() + " "; // Mengambil kata dari stack
        kataStack.pop(); // Menghapus kata dari stack
    }

    return kalimatTerbalik;
}
```

```
int main() {
    string kalimat;
    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, kalimat);

    // Memastikan kalimat memiliki minimal 3 kata
    int jumlahSpasi = 0;
    for (char c : kalimat) {
        if (c == ' ') {
            jumlahSpasi++;
        }
    }

    if (jumlahSpasi < 2) {
        cout << "Kalimat tidak memiliki minimal 3 kata!" << endl;
        return 1;
    }

    // Memanggil fungsi untuk membalikkan kalimat
    string kalimatTerbalik = reverseSentence(kalimat);

    // Menampilkan kalimat yang sudah dibalik
    cout << "Kalimat setelah dibalik: " << kalimatTerbalik << endl;

    return 0;
}
```

Screenshots output:



Deskripsi :

1. **reverseSentence(string kalimat)**: Fungsi ini menerima sebuah string **kalimat** dan mengembalikan string yang merupakan kalimat yang sudah dibalik. Pada fungsi ini:
    - Dideklarasikan sebuah stack **kataStack** untuk menyimpan setiap kata dalam kalimat.
    - Kalimat dipisahkan menjadi kata-kata dan setiap kata dimasukkan ke dalam stack.
    - Setelah semua kata dimasukkan ke dalam stack, kata-kata diambil kembali dari stack dan digabungkan untuk membentuk kalimat baru yang sudah dibalik.

2. **main()**: Fungsi utama program. Pada fungsi ini:
    - Pengguna diminta untuk memasukkan sebuah kalimat dengan minimal 3 kata.
    - Kalimat yang dimasukkan oleh pengguna kemudian disimpan dalam variabel **kalimat**.
    - Jumlah spasi dalam kalimat dihitung untuk memastikan kalimat memiliki minimal 3 kata.
    - Jika kalimat memiliki kurang dari 3 kata, program akan menampilkan pesan kesalahan.
    - Jika kalimat memiliki minimal 3 kata, program akan memanggil fungsi **reverseSentence()** untuk membalikkan kalimat.
    - Kalimat yang sudah dibalik kemudian ditampilkan kepada pengguna.

E. Referensi

https://medium.easyread.co/memahami-konsep-stack-secara-sederhana-bd4409ec560c