# PERTEMUAN 5

# STRUKTUR DATA

# STACK

**Disusun oleh:**

**Rama Pramudya Wibisana 2022320019**

**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS INFORMATIKA**

**UNIVERSITAS BINA INSANI**

**BEKASI**

**2022**

- **PENJELASAN SYNTAX**

```
#include <iostream>
#include <conio.h>
#define maxstack 4
using namespace std;
```

**`#include <iostream>`** berfungsi untuk mengimpor fungsi-fungsi yang sudah didefinisikan pada *header file.*

**`#include <conio.h>`** berfungsi untuk menampilkan hasil antarmuka kepada pengguna. Fungsi yang terdapat dalam conio.h adalah getch(), getche(), dan clrscr().

**`#define maxstack 4`** berfungsi untuk menentukan maksimal pada stack adalah 4 data.

**`using namespace std;`** berarti kita menuliskan perintah 'c++, saya ingin gunakan semua yang ada dalam namespace std (standart), seperti cin, cout, endl, vector, string, pair, map, queue, deque, dll yang merupakan fitur standard pada library C++.'

```
struct STACK
{
    int top;
    string data[4];
};
string data;

struct STACK newstack;
```

**`Struct STACK`** berarti saya ingin menggunakan tipe data **`struct`** yang identitasnya ialah **`STACK.`**

Untuk variable **`top`** saya menggunakan tipe data **`int.`**

Untuk variable **`data`** saya menggunakan tipe data **`string`** yang memiliki 4 index.

Struct **`STACK`** dijadikan suatu tipe data, dimana disebut tipe data abstrak.

```
bool isfull()
{
    if (newstack.top == maxstack)
        return true;
    else
        return false;
}
```

Berfungsi untuk mengetahui apakah suatu stack sedang penuh adalah dengan membandingkan **`newstack.top`** dengan **`maxstack`**, jika kondisi true maka stack dalam posisi penuh, dan sebaliknya.

```
23  bool isempty()
24  {
25      if (newstack.top == -1)
26          return true;
27      else
28          return false;
29  }
```

Befungsi untuk mengetahui apakah suatu stack dalam keadaan kosong adalah dengan membandingkan **`newstack.top`** dengan -1, jika kondisi true maka stack dalam posisi kosong, dan sebaliknya.

```
31  void push(string data)
32  {
33      if (isfull() == true)
34      {
35          puts("\nKeranjang belanja sudah penuh!\n");
36          system("pause");
37      }
38      else
39      {
40          newstack.top++;
41          newstack.data[newstack.top] = data;
42      }
43  }
```

Pertama kali akan dicek apakah stack dalam keadaan penuh, jika true maka akan tercetak string pada layar “Keranjang belanja sudah penuh!”. Jika bernilai false maka **`newstack.top`** akan diincrement kemudian data yang tadi diinputkan ditambahkan pada stack.

```
45  void pop()
46  {
47      if (isempty() == true)
48      {
49          cout << "\nKeranjang belanja kosong!\n";
50      }
51      else
52      {
53          cout << "\nBarang yang dibatalkan : " << newstack.data[newstack.top] << endl;
54          newstack.top--;
55      }
56  }
```

Pertama kali yang akan dilakukan program adalah mengecek apakah stack dalam keadaan kosong, jika true maka akan tercetak string pada layar “Keranjang belanja kosong!". Jika bernilai false maka data pada posisi teratas akan diambil, dan kemudian nilai **`newstack.top`** didecrement sehingga posisi teratas pada stack berganti dengan data di bawah top sebelumnya.

```
58  void print()
59  {
60      printf("\nBarang yang anda masukkan ke keranjang : \n");
61      printf("=============================================\n");
62      for (int x = 0; x <= newstack.top; x++)
63      {
64          cout << newstack.data[x] << "  |  ";
65      }
66  }
```

Dengan memanfaatkan perulangan for, fungsi ini akan mencetak seluruh data yang berada di dalam stack.

```
68  void clear()
69  {
70      newstack.top = -1;
71      printf("\nKeranjang belanja kosong!\n");
72  }
```

Saat fungsi ini dipanggil maka posisi **`newstack.top`** diinisialisasi berada pada -1. Seperti halnya mereset ulang suatu stack yang membuat isinya akan hilang.

```
74  int main()
75  {
76      newstack.top = -1;
77
78      char menu;
79      char kembali;
```

**`newstack.top`** diinisialisasi berada pada -1.

Variabel **`menu`** sebagai tipe data **`char`**.

Variabel **`kembali`** sebagai tipe data **`char`**.

```
81  do
82  {
83      system("cls");
84      printf("\t     E-COMMERCE\n");
85      printf("\t====================\n\n");
86      printf("Menu : \n");
87      puts("1. Barang yang akan dibeli");
88      puts("2. Keranjang");
89      puts("3. Batalkan barang yang akan dibeli");
90      puts("4. Bersihkan keranjang");
91      puts("5. Check Out");
92      puts("6. Exit");
93
94      cout << "\nPilih Menu : ";
95      cin >> menu;
```

Masuk ke perulangan do-while dengan kondisi ketika

**`ulang == 'y' || ulang 'Y'`**

Pada perulangan ini akan ditampilkan 6 menu pilihan yang dapat dipilih oleh user.

Kemudian ada input yang menggunakan variabel **`menu`**.

```
97  if (menu == '1')
98  {
99      cout << "\nMasukan barang : ";
100     cin >> data;
101     push(data);
102     kembali = 'y';
103 }
```

Jika input **menu == '1'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi pertama yaitu fungsi **push**.

```
104 else if (menu == '2')
105 {
106     if (isempty() == false)
107     {
108         print();
109         cout << "\n\nKembali? (y/t) ";
110         cin >> kembali;
111     }
112     else
113     {
114         clear();
115         cout << "\nKembali? (y/t) ";
116         cin >> kembali;
117     }
118 }
```

Jika input **menu == '2'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi kedua.

Kemudian program akan mengecek apakah stack dalam keadaan kosong, jika false maka program akan memanggil fungsi **clear()** dan akan tercetak string pada layar “Keranjang belanja anda kosong". Jika bernilai true maka program akan memanggil fungsi **print()** yang nantinya akan dicetak.

```
119 else if (menu == '3')
120 {
121     pop();
122     kembali = 'y';
123     getch();
124 }
```

Jika input **menu == '3'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi ketiga.

```
125 else if (menu == '4')
126 {
127     clear();
128     cout << "\n\nKembali? (y/t) ";
129     cin >> kembali;
130 }
```

Jika input **menu == '4'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi keempat yaitu fungsi **pop**.

```
131  else if (menu == '5')
132  {
133      if (isempty() == false)
134      {
135          cout << "Barang berhasil di-check out\n";
136          clear();
137          cout << "\n\nKembali? (y/t) ";
138          cin >> kembali;
139      }
140      else
141      {
142          cout << "Tidak ada barang di keranjang\n";
143          cout << "\nKembali? (y/t) ";
144          cin >> kembali;
145      }
146  }
```

Jika input **menu == '5'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi kelima.

Kemudian program akan mengecek apakah stack dalam keadaan kosong, jika false maka program akan menampilkan print "Tidak ada barang di keranjang". Jika bernilai true maka program akan menampilkan print "Barang berhasil di-check out" dan memanggil fungsi **clear()** yang nantinya akan dicetak.

```
147  else if (menu == '6')
148  {
149      exit(0);
150  }
```

Jika input **menu == '6'** maka program akan menjalankan seluruh pernyataan yang ada pada kondisi keenam yaitu exit program.

```
151      } while (kembali == 'Y' || kembali == 'y');
152  }
```

- **OUTPUT SYNTAX**

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu :
```

*Tampilan utama*

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu : 1

Masukan barang :
```

*Menu 1 push stack*

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu : 1

Masukan barang : celana

Keranjang belanja sudah penuh!

Press any key to continue . . .
```

*Menu 1 apabila stack penuh*

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu : 2

Barang yang anda masukkan ke keranjang :
=============================================
baju  |  buku  |  komik  |  sepatu  |  kemeja  |

Kembali? (y/t)
```

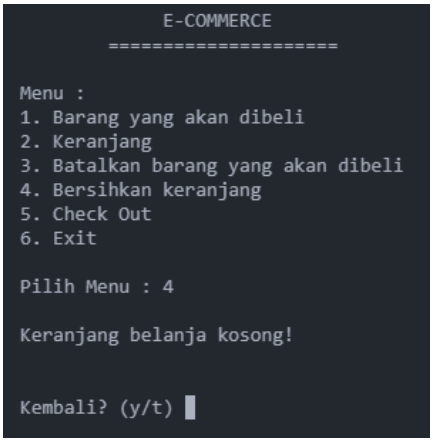
*Menu 2 apabila stack terisi*

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu : 2

Keranjang belanja kosong!


Kembali? (y/t)
```

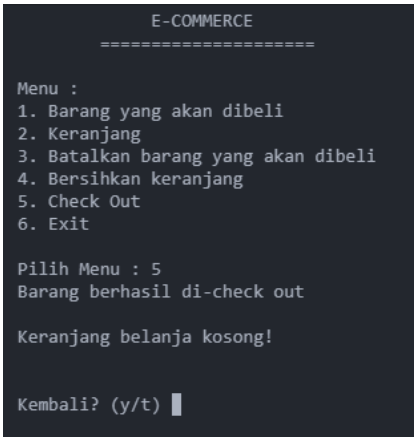
*Menu 2 apabila stack kosong*

```
            E-COMMERCE
        ====================

Menu :
1. Barang yang akan dibeli
2. Keranjang
3. Batalkan barang yang akan dibeli
4. Bersihkan keranjang
5. Check Out
6. Exit

Pilih Menu : 3

Barang yang dibatalkan : kemeja
```

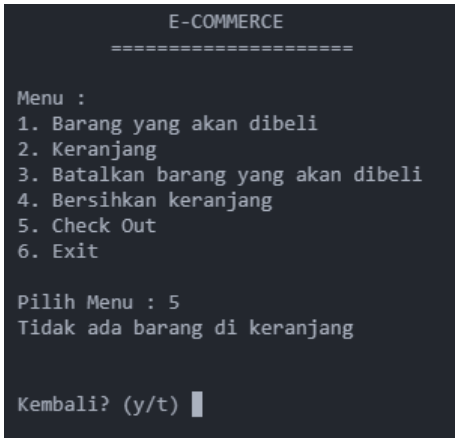
*Menu 3 pop stack*



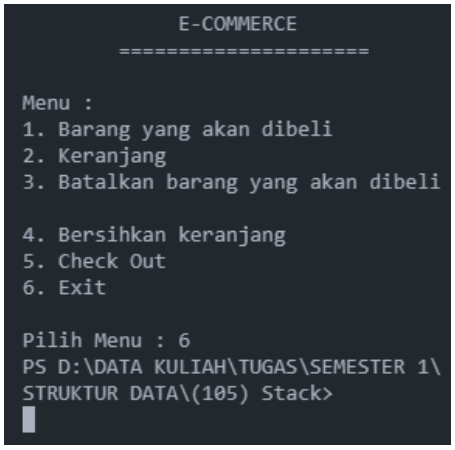

*Menu 4*




*Menu 5 apabila stack terisi*




*Menu 5 apabila stack kosong*




*Menu 6 exit*