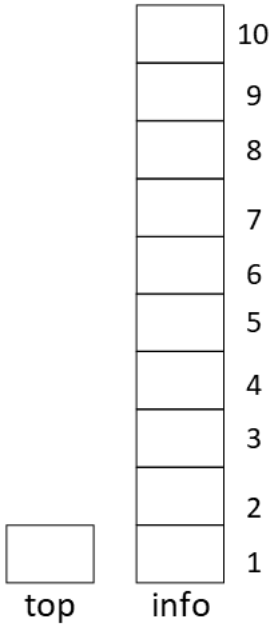
Praktikum Modul ke-7: Stack

Dalam praktikum ini Anda diminta untuk membuat stack dengan representasi array. Buatlah ADT stack pada file “stack_character.h” dengan struktur berikut:

Notasi Algoritmik	Bahasa C++

Constant MAXSTACK : integer = 10
type infotype : char
type stack :
<
info : array[1.. MAXSTACK] of char,
top : integer
>
	const int MAXSTACK = 10;
	typedef char infotype;
	struct stack {
	infotype info[MAXSTACK];
	int top;
	};

Ilustrasi untuk struktur tersebut adalah sebagai berikut



Elemen info[1.. MAXSTACK] akan digunakan untuk menampung karakter. Sedangkan Top akan digunakan untuk menampung informasi mengenai indeks elemen info yang paling atas. Jadi, top = 0 artinya stack dalam keadaan kosong, top = 1, artinya stack berisi satu elemen.

Gunakan alias sebagai berikut:

top(S) untuk S.top

info(S) untuk S.info

Prosedur dan fungsi yang digunakan adalah sebagai berikut

Notasi Algoritmik	Bahasa C++
procedure createStack(input/output S : stack) { I.S. - F.S. terbentuk stack dengan Top = 0 }	void createStack(stack &S)
function isEmpty(S : stack): boolean { Mengembalikan nilai true jika stack kosong }	bool isEmpty(stack S)
function isFull(S : stack): boolean { Mengembalikan nilai true jika stack penuh }	bool isFull(stack S)
procedure push(input/output S : stack, input xpush : infotype) { I.S. mungkin penuh F.S. menambahkan elemen pada stack dengan nilai xpush, top = top + 1}	void push(stack &S, infotype xpush)
procedure pop(input/output S : stack, output xpop: infotype) { I.S. mungkin kosong F.S. mengeluarkan elemen pada stack dengan nilai xpop, top = top - 1}	void pop(stack &S, infotype &xpop)
procedure printInfo(input S : stack) { I.S. stack mungkin kosong F.S. Jika stack tidak kosong, maka menampilkan semua info yang ada pada stack }	void printInfo(stack S)

Selanjutnya buat implementasi ADT stack pada file “stack_character.cpp”

Notasi Algoritmik	Bahasa C++

procedure createStack(input/output S : stack)
{ I.S. -
F.S. terbentuk stack dengan Top = 0 }
Kamus:
-
Algoritma:
Top(S) ← 0
	void createStack(stack &S){
	
	/* Lengkapi kode */
	
	}

Notasi Algoritmik	Bahasa C++

function isEmpty(S : stack): boolean
{ Mengembalikan nilai true jika stack kosong }
Kamus:
  -
Algoritma:
if Top(S) = 0 then
  → true
else
  → false
{ end if}
	bool isEmpty(stack S)){
	
	  /* Lengkapi kode */
	
	}

Notasi Algoritmik	Bahasa C++

function isFull(S : stack): boolean
{ Mengembalikan nilai true jika stack penuh }
Kamus:
  -
Algoritma:
if Top(S) = MAXSTACK then
  → true
else
  → false
{ end if}
	bool isFull(stack S)){
	
	  /* Lengkapi kode */
	
	}

Notasi Algoritmik	Bahasa C++

procedure push(input/output S : stack, input xpush : infotype)
{ I.S. mungkin penuh
F.S. menambahkan elemen pada stack dengan nilai xpush, top = top + 1}
 	
Kamus:
  -
Algoritma:
if isFull(S) = false then
  Top(S) ← Top(S) + 1
  Info(S)[Top(S)] ← x
else
  Output(“Stack penuh”)
{ end if}
	void push(stack &S, infotype xpush)){
	
	  /* Lengkapi kode */
	
	}

Notasi Algoritmik	Bahasa C++

procedure pop(input/output S : stack, output xpop: infotype)
{ I.S. mungkin kosong
F.S. mengeluarkan elemen pada stack dengan nilai xpop, top = top - 1}
Kamus:
-
Algoritma:
if isEmpty (S) = false then
xpop ← Info(S)[Top(S)]
Top(S) ← Top(S) - 1
else
Output("Stack kosong")
{ end if}
	void pop(stack &S, infotype &xpop)){
	
	/* Lengkapi kode */
	
	}

Notasi Algoritmik	Bahasa C++

procedure printInfo(input S : stack)
{ I.S. stack mungkin kosong
F.S. Jika stack tidak kosong, maka menampilkan semua info yang ada pada stack }
Kamus:
i: integer
Algoritma:
if isEmpty (S) = false then
for i ← Top(S) downto 1 do
output(info(S)[i])
{ end for }
else
Output("Stack kosong")
{ end if}
	void printInfo(stack S)){
	
	/* Lengkapi kode */
	
	}

Selanjutnya, buatlah procedure ascending dan procedure descending yang masing-masing membuat info stack terurut menaik dan menurun.

```
void ascending(stack &S);
void descending(stack &S);
```

Kemudian, buat procedure stringToStack untuk menerima masukkan berupa string yang setiap karakternya akan diinputkan ke dalam stack.

```
void stringToStack(stack &S, const string &str);
```

Setelah itu, buat procedure reverseStack untuk membalik urutan elemen dalam suatu stack.

```
void reverseStack(stack &S);
```

Lakukan pengujian terhadap implementasi stack! Buat kode pada “main.cpp” sebagai berikut (Silakan copy paste):

```
stack S1, S2, S3;
infotype xpop;

createStack(S1);
printInfo(S1);

push(S1, 'E');
printInfo(S1);

push(S1, 'T');
printInfo(S1);

push(S1, 'U');
printInfo(S1);

pop(S1, xpop);
printInfo(S1);

cout << endl << "Data diurutkan ascending:" << endl;
ascending(S1);
printInfo(S1);
cout << endl << "Data diurutkan descending:" << endl;
descending(S1);
printInfo(S1);

createStack(S2);
string kalimat = "Hai kamu!!!";
cout <<endl<< "\'Hai kamu!!!\' di dalam stack:" <<endl;
stringToStack(S2,kalimat);
printInfo(S2);
cout << endl << "Data diurutkan ascending:" << endl;
ascending(S2);
printInfo(S2);
cout << endl << "Data diurutkan descending:" << endl;
descending(S2);
printInfo(S2);

createStack(S3)
cout <<endl<< "Membalik urutan elemen " <<endl;
string input;
```

```
    cout << "Masukkan string: ";
    // Inputkan “IFLAB 2024/2025”
    getline(cin, input); // Menggunakan getline untuk membaca string dengan spasi

    stringToStack(S3, input);
    cout << endl << "Data sebelum dibalik:";
    printInfo(S3);
    cout << endl;

    reverseStack(S3)
    cout << "Data  setelah dibalik:";
    printInfo(S3);
    cout << endl;

    return 0;
}
```