Tugas Pendahuluan Modul 7

STRUKTUR DATA – Ganjil 2025/2026

Stack

Ketentuan Tugas Pendahuluan

1. Tugas Pendahuluan dikerjakan secara Individu.
2. TP ini bersifat WAJIB, tidak mengerjakan = PENGURANGAN POIN JURNAL / TES ASESMEN.
3. Hanya MENGUMPULKAN tetapi TIDAK MENGERJAKAN = PENGURANGAN POIN JURNAL / TES ASESMEN.
4. Deadline pengumpulan TP Modul 7 adalah Senin, 27 Oktober 2025 pukul 06.00 WIB.
5. Tidak ada TOLERANSI KETERLAMBATAN. TERLAMBAT atau TIDAK MENGUMPULKAN TP maka DIANGGAP TIDAK MENGERJAKAN.
6. DILARANG PLAGIAT (PLAGIAT = E).
7. Kerjakan TP dengan jelas agar dapat dimengerti.
8. Untuk setiap soal nama fungsi atau prosedur WAJIB menyertakan NIM, contoh:

```
int namaFungsi_10301XXXXXXXX(...);
```

9. File diupload di LMS menggunakan format PDF dengan format:

```
TP_MOD_[XX]_NIM_NAMA.pdf
```

CP (WhatsApp):

- Falih (+62 857-2774-5199)
- Faried (+62 813-8921-4045)
- Arief (+62 852-1252-8394)
- Evan (+62 896-0298-0999)

SELAMAT MENGERJAKAN!

1 Menghitung Jumlah Elemen dalam Stack

Diberikan kumpulan elemen bilangan bulat dalam stack. Anda diminta untuk membuat program dengan mengimplementasikan konsep stack, yang menjumlahkan semua elemen yang ada di dalam stack dan mengembalikan hasil penjumlahan tersebut.

1.1 ADT Program

Buatlah ADT Stack (`stack.h`), serta dengan subprogram primitif nya.

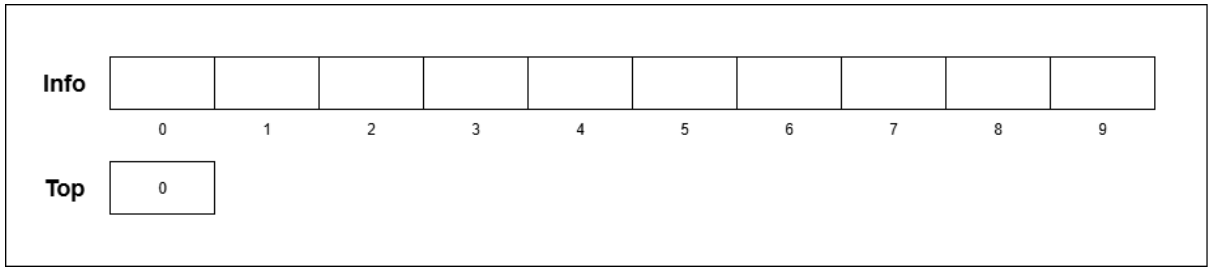
```
constant MAXSTACK : integer = 10

type infotype : integer

type stack <
    info : array[0..MAXSTACK-1] of infotype
    top : integer
>

procedure createStack(in/out S : stack)
function isEmpty(S : stack) -> boolean
function isFull(S : stack) -> boolean
procedure push(in/out S : stack, in x : infotype)
function pop(S : stack) -> infotype
function sumStack(S : stack) -> integer
```

Ilustrasi untuk struktur di atas:



Elemen info[0..MAXTSACK-1] akan digunakan untuk menampung data elemen, 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.

1.2 Implementasi Subprogram

Buatlah subprogram (`stack.cpp`) untuk stack nya.

```
procedure createStack(in/out S : stack)
{I.S. Tersedia stack S sembarang.
```

```
 F.S. Terdefinisi stack S dengan top = 0.}
algorithm
    S.top <- 0
endprocedure

function isEmpty(S : stack) -> boolean
{I.S. Terdefinisi stack S.
 F.S. Mengembalikan true jika S kosong, dan false jika sebaliknya.}
algorithm
    return S.top == 0
endfunction

function isFull(S : stack) -> boolean
{I.S. Terdefinisi stack S.
 F.S. Mengembalikan true jika S penuh, dan false jika sebaliknya.}
algorithm
    return S.top == MAXSTACK
endfunction

procedure push(in/out S : stack, in x : infotype)
{I.S. Terdefinisi stack S.
 F.S. Menambahkan elemen pada stack dengan nilai x, dengan top di
tambahkan dengan 1. Jika S penuh, maka x tidak di sisipkan.}
algorithm
    if (not isFull(S)) then
        S.info[S.top] <- x
        S.top <- S.top + 1
    endif
endprocedure

function pop(S : stack) -> infotype
{I.S. Terdefinisi stack S yang tidak kosong.
 F.S. Mengembalikan nilai elemen pada stack S pada indeks ke `S.top`,
dengan top di kurangkan dengan 1.}
dictionary
    x : infotype

algorithm
    S.top <- S.top - 1
    x <- S.info[S.top]
    return x
endprocedure

function sumStack(S : stack) -> integer
{I.S. Terdefinisi stack S.
 F.S. Mengembalikan jumlah dari seluruh elemen dalam S.}
dictionary
    total : integer
algorithm
    total <- 0

    while (not isEmpty(S)) do
        total <- total + pop(S)
    endwhile

    return total
endfunction
```

1.3 Main Program

Implementasikan program utama seusai dengan teks dibawah (teks bold bergaris bawah adalah input).

```
input angka ke-1: 1
input angka ke-2: 2
input angka ke-3: 3
input angka ke-4: 4
input angka ke-5: 5
input angka ke-6: 6
input angka ke-7: 7
input angka ke-8: 8
input angka ke-9: 9
input angka ke-10: 10
stack sudah penuh!

hasil penjumlahan elemen dalam stack: 55
```