

# Pemahaman Modul

## Praktikum Struktur Data.

Kode\_Asi sten\_PJ\_: \_MTN

Kel ompok = A11.4301U

NIM : A11.2012.06758

NIM : A11.2012.06648

Nama : Muhammad Adhi D

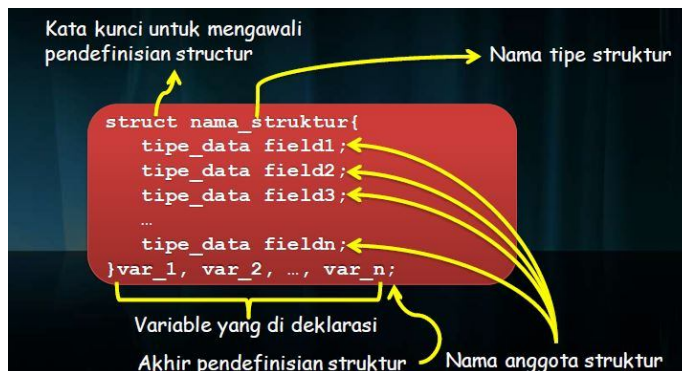
Nama : Agustinus Henry P

### PSDA – 01 (Stack Statis dan Dinamis).

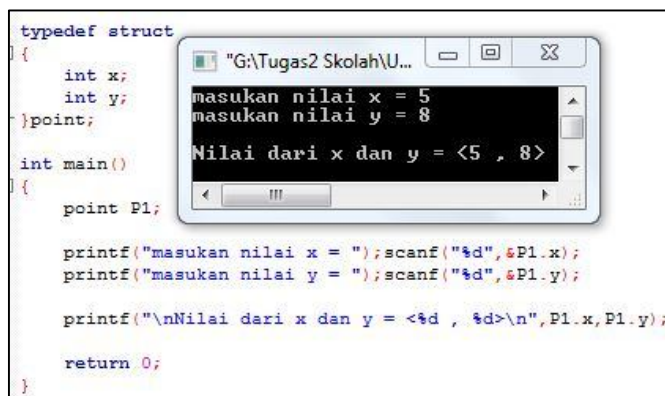
#### A. Pendahuluan Mengenai ADT.

Struktur merupakan type data yang dapat menyimpan sejumlah data yang berytpe berbeda, misalkan di definisan ADT identitas mahasiswa, maka di dalam struktur tersebut terdapat beberapa variabel berbeda type, misalnya "char" sebagai nama, "int/float" sebagai nilai dll.

Dalam bahasa C kita dapat menggunakan kata kunci **"struct"**, yang diikuti block **{}** dan diakhiri dengan tanda semi colon **;**, berikut definisi struktur dalam bahasa C.



Contoh program sederhana sebagai berikut.



Di berikan sebuah ADT dengan type datanya adalah integer yaitu **int** : x, dan **int** : y, dengan nama "point".

# Pemahaman Modul

## Praktikum Struktur Data.

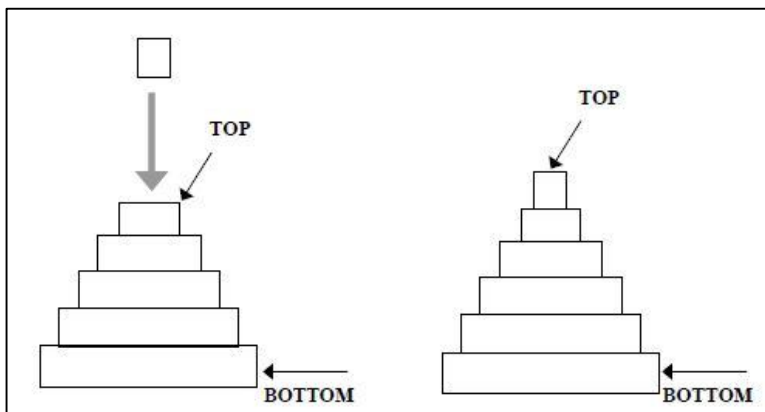
Dari contoh di atas pertama harus dilakukan pendeklarasian variabel ADT dengan nama point, diberi nama **"P1"**. Untuk mengakses variabel yang ada pada ADT (x dan y), dengan cara diberi tanda penghubung yaitu titik **"."**, misal akan mengakses variabel x dari variabel P1, maka ditulis P1.x begitu juga dengan variabel y.

### B. Definisi Stack

Stack merupakan "List Linier" yang dikenali elemen puncaknya (TOP). Aturan penyisipan dan penghapusan elemen yaitu :

- Penyisipan selalu dilakukan di atas TOP.
- Penghapusan selalu dilakukan pada TOP.

Elemen stack tersusun secara LIFO (*Last In First Out*). Dengan definisi ini maka representasi table linier sangat tepat untuk mewakili stack karena operasi penambahan dan pengurangan hanya dilakukan salah satu ujung table.



### C. Stack Statik

Stack Statik merupakan stack dimana ukuran kapasitas maksimumnya sudah ditentukan dari awal pembuatan program, dan pendefinisian variabel stack bertipe **"array"**.

- ADT bentukan bertipe integer, diberi nama Infotype dan address.
- Type data penampung variabel dalam stack adalah integer array, dengan ukuran idxmax+1.
- TOP merupakan alamat dari index suatu "array T[]".

```
#define idxmax 10
#define nil 0
typedef int InfoType;
typedef int address;
typedef struct
{
    InfoType T[idxmax+1];
    address TOP;
}Stack;
```

# Pemahaman Modul

## Praktikum Struktur Data.

### C.1 Selektor.

```
#define TOP(S) (S).TOP  
#define InfoTop(S) (S).T[(S).TOP]
```

Definisi selector, "S" merupakan sebuah stack dengan,

- ✓ **TOP(S)** adalah alamat dari elemen TOP, dimana operasi penyisipan dan penghapusan dilakukan.
- ✓ **InfoTOP(S)** adalah informasi suatu nilai yang disimpan dalam elemen, dengan alamat posisi TOP.
- ✓ Saat pendefinisian stack kosong, dilakukan inisialisasi nilai pada variabel TOP(S), yaitu bernilai **NIL** atau **0**.

### C.2 Definisi Fungsional.

```
void createempty (Stack *St);  
boolean isempty (Stack St);  
boolean isfull (Stack St);  
void push (Stack *St, int x);  
void pop (Stack *St, int *x);
```

#### a. Void CreateEmpty.

Membuat stack kosong, dengan parameter pointer (\*St), bertipe Stack dan berkapasitas MaxEl, dengan index antara 1~MaxEl, indeks 0 tidak digunakan, dengan ciri stack kosong adalah nilai dari TOP(S) bernilai NIL.

{**I.S** = Sembarang kondisi.}

{**F.S** = membuat stack kosong berkapasitas MaxEl.}

Algoritma.

**TOP(S) <-- NIL.**

{inisialisasi TOP(S) dengan nilai 0}.

#### b. Void Push.

Menambahkan elemen stack \*St, dengan nilai integer x, \*ST dan x adalah parameter dari prosedur "Push".

{**I.S** = kemungkinan kosong, dengan kapasitas penampung elemen tidak kosong.}

{**F.S** = nilai x menjadi TOP yang baru, dan alamat TOP nilainya akan bertambah 1.}

# Pemahaman Modul

## Praktikum Struktur Data.

---

### Algoritma

- Jika stack tidak penuh, maka tambahkan elemen x ke dalam elemen penampung beralamat TOP(S).
- Kemudian alamat dari TOP(S), bertambah 1.

**TOP(\*S) <--- TOP(\*S) + 1;**

**InfoTop(\*St) <--- x;**

### c. Void Pop.

Mengambil dan menghapus nilai dari Elemen InfoTop(S), dengan kondisi alamat TOP(S). Kemudian menyimpannya pada variabel tertentu. Terdapat 2 parameter yaitu Stack \*S dan Integer \*x.

{**I.S** = Stack tidak mungkin kosong.}

{**F.S** = nilai dari TOP(S) akan hilang, dan index TOP(S) berkurang 1.}

### Algoritma.

- Jika stack tidak kosong, maka ambil nilai dari InfoTOP(S) beralamatkan index TOP(S).
- Kemudian alamat dari TOP(S), berkurang 1.

**\*x <--- InfoTOP(\*S);**

**TOP(\*S) <-- TOP(\*S) -1;**

### d. Boolean IsEmpty.

Mengecek kondisi dari Stack S, apakah stack tersebut kosong?.

### Algoritma.

- Jika kosong maka bernilai **"TRUE"**, selain itu **"FALSE"**.

**Return TOP(S) == NIL.**

### e. Boolean IsFull.

Mengecek kondisi dari Stack S, apakah stack tersebut Penuh?.

### Algoritma.

- Jika penuh maka bernilai **"TRUE"**, selain itu **"FALSE"**.

**Return TOP(S) == MaxEL.**

# Pemahaman Modul

## Praktikum Struktur Data.

### D. Stack Dinamik

Stack dinamik merupakan sebuah stack dimana ukuran kapasitasnya dapat ditentukan secara bebas saat program mulai dieksekusi. Type data dari stack dinamik bukanlah berupa array, namun **"pointer"**.

- struktur dari ADT stack hampir sama dengan stack static, perbedaanya hanya terletak pada type data elemen penampung yaitu integer dengan "Pointer". Dan tambahan variabel baru yaitu Size.

```
#define NIL 0
typedef int infotype;
typedef int address;
typedef struct
{
    infotype *T;
    address TOP;
    int size;
}Stack;
```

#### D.1 Selektor

```
#define TOP(S) (S).TOP
#define Size(S) (S).size
#define InfoTop(S) (S).T[(S).TOP]
```

Definisi selector, "**S**" merupakan sebuah stack dengan,

- ✓ **TOP(S)** adalah alamat dari elemen TOP, dimana operasi penyiipan dan penghapusan dilakukan.
- ✓ **InfotOP(S)** adalah informasi suatu nilai yang disimpan dalam elemen, dengan alamat posisi TOP.
- ✓ **Size(S)** adalah ukuran dari daya tampung stack dinamis.

#### D.2 Definisi Fungsional

```
void createempty (Stack *St,int sz);
boolean isempty (Stack st);
boolean isfull (Stack st);
void poop (Stack *St , int *x);
void push (Stack *St , int x);
void Dealokasi (Stack *St);
```

##### a. Void CreateEmpty

```
(*St).T = (int *) malloc(sz * sizeof(int));
TOP(*St) = NIL;
Size(*St) = sz;
```

a

Procedure ini hampir sama dengan stack static, hanya bedanya ada pada pengalokasian memori saat program berjalan, hal ini ditandai dengan keyword **"malloc"** yang berarti memori allocation. Beserta inisialisasi nilai variabel ke dalam variabel Size(\*S).

##### b. Void Dealokasi.

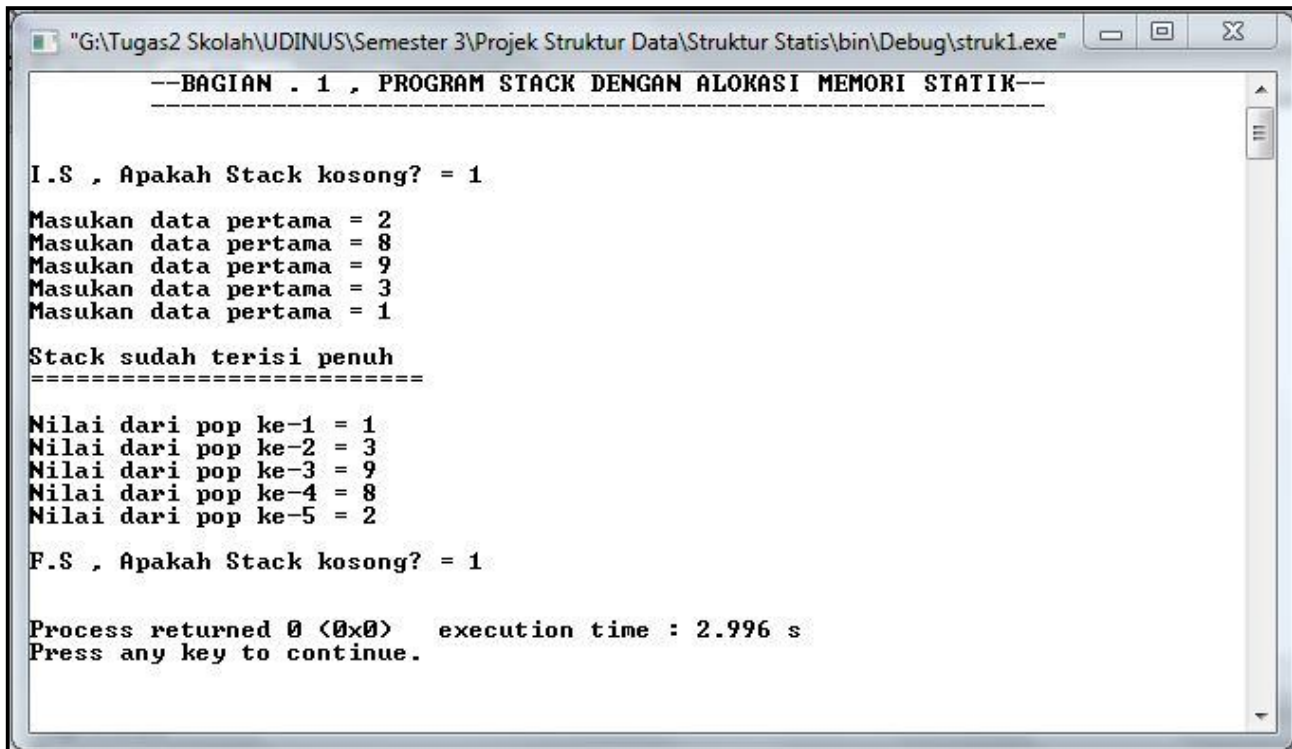
```
Size(*St) = NIL;
free((*St).T);
```

Prosedure ini berfungsi sebagai dealokasi seluruh table memori sekaligus. Ditandai dengan keyword "free". Ukuran stack di set menjadi NIL atau 0, dan semua data pada stack akan di hapus.

# Pemahaman Modul

## Praktikum Struktur Data.

### A. Contoh Program STACK STATIS.



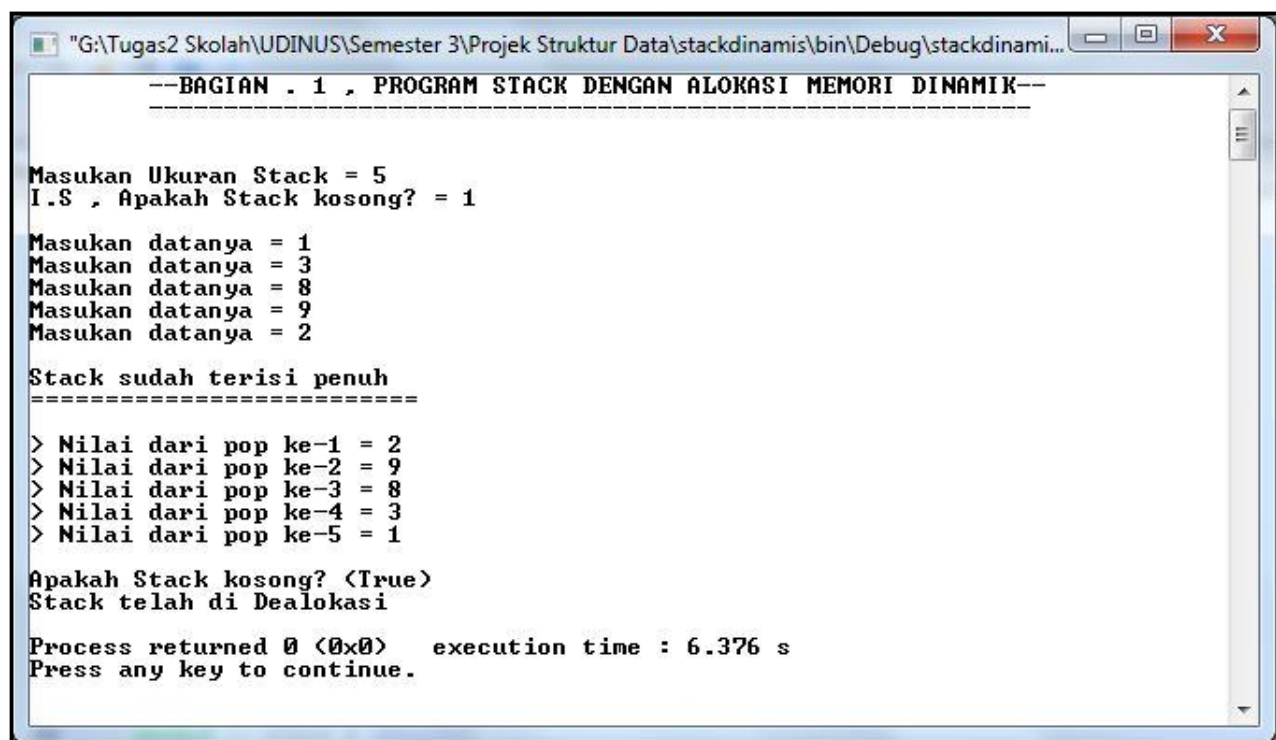
```
"G:\Tugas2 Sekolah\UDINUS\Semester 3\Projek Struktur Data\Struktur Statis\bin\Debug\struk1.exe"

--BAGIAN . 1 , PROGRAM STACK DENGAN ALOKASI MEMORI STATIS--

I.S , Apakah Stack kosong? = 1
Masukan data pertama = 2
Masukan data pertama = 8
Masukan data pertama = 9
Masukan data pertama = 3
Masukan data pertama = 1
Stack sudah terisi penuh
=====
Nilai dari pop ke-1 = 1
Nilai dari pop ke-2 = 3
Nilai dari pop ke-3 = 9
Nilai dari pop ke-4 = 8
Nilai dari pop ke-5 = 2
F.S , Apakah Stack kosong? = 1

Process returned 0 (0x0)   execution time : 2.996 s
Press any key to continue.
```

### B. Contoh Program STACK DINAMIS.



```
"G:\Tugas2 Sekolah\UDINUS\Semester 3\Projek Struktur Data\stackdinamis\bin\Debug\stackdinami...

--BAGIAN . 1 , PROGRAM STACK DENGAN ALOKASI MEMORI DINAMIS--

Masukan Ukuran Stack = 5
I.S , Apakah Stack kosong? = 1
Masukan datanya = 1
Masukan datanya = 3
Masukan datanya = 8
Masukan datanya = 9
Masukan datanya = 2
Stack sudah terisi penuh
=====
> Nilai dari pop ke-1 = 2
> Nilai dari pop ke-2 = 9
> Nilai dari pop ke-3 = 8
> Nilai dari pop ke-4 = 3
> Nilai dari pop ke-5 = 1
Apakah Stack kosong? <True>
Stack telah di Dealokasi

Process returned 0 (0x0)   execution time : 6.376 s
Press any key to continue.
```