**TM 03 : ADT Array dan Struct**

*Array* adalah suatu *ADT*. Seperti halnya ADT lain, *array* mempunyai data anggota dan operasi, metode atau fungsi anggota. Elemen *array* diakses berdasarkan posisinya di dalam *array* relatif terhadap elemen pertama. Operasi utama *array* adalah *store* dan *retrieve*. Operassi *store* untuk menyimpan elemen e di indeks i array A[] adalah A[i] = e sedangkan operasi *retrieve* untuk mengambil suatu elemen dari array A[] di indeks i dan disimpan di elemen e adalah e = A[i].

1. Desain *class Array* berdasarkan uraian *ADT Array* di atas dengan ketentuan level akses data anggota adalah *private* dan level akses fungsi anggota adalah *public*. Selain itu jangan lupa untuk menambahkan *constructor* dan *destructor* pada *class* tersebut. Dalam desain ini, semua fungsi anggota kecuali *constructor* dan *destructor*, harus berbentuk **prototype**.

const int MAX =10;

class Array {

private :

 int A[MAX};

 int i, e;

public :

 Array() { for ( i =0; i < MAX; i++) A[i] = 0; }

 ~Array();

 void store( int, int );

 int retrieve (int, int);

};

1. Selanjutnya definisikan semua fungsi anggota di luar *class Array*. Ingat jangan lupa untuk menggunakan *scope resolution operator* ::

void Array :: store (int i, int e) { A[i] = e; }

int Array :: retrieve (int i, int e) { e = A[i]; }

1. Terakhir, tulis fungsi *main()* sebagai *driver* untuk menguji desain kamu.

void main() {

int e;

Array a;

 a.store(1,11);

 a.store(2,22);

 a.store(3,33);

 **cout** << endl << a,retrieve (3,e);

 **cout** << endi << a.retrieve(2,e);

 **cout** << endl << a.retrieve(1,e);

}

Seperti halnya *array*, *struct* adalah suatu ADT yangmempunyai data anggota dan operasi, metode atau fungsi anggota. Elemen, *membe*r atau *field* *struct*  diakses menggunakan operator *dot* atau *pointe*r ->. Bentuk umumnya adalah sebagai berikut:

<structure\_variabel>.<field\_name>

<structure\_pointer> → <field\_name>

Operasi utama di *struct* adalah *store(e)* dan *retrieve()*. Operassi *store(e)* untuk menyimpan elemen e di *field* i *struct* s adalah s.i = e sedangkan operasi *retrieve()* untuk mengambil suatu elemen dari *struct* s di field i dan disimpan di elemen e adalah e = s.i.

1. Diketahui struktur *dtMhs* dengan tiga data anggota : nim, nama dan IPK serta fungsi anggota *store()* dan *retrieve().* Desain *class dtMhs* berdasarkan uraian di atas dengan ketentuan *level* akses data anggota adalah *private* dan *level* akses fungsi anggota adalah *public*. Selain itu jangan lupa untuk menambahkan *constructor* dan *destructor* pada *class* tersebut. Dalam desain ini, semua fungsi anggota kecuali *constructor* dan *destructor*, harus berbentuk **prototype**.
2. Selanjutnya definisikan semua fungsi anggota di luar *class Struct* Ingat jangan lupa untuk menggunakan *scope resolution operator* ::
3. Terakhir, tulis fungsi *main()* sebagai *driver* untuk menguji desain kamu.

**I AM PROUD AS A BINUSIAN**

a.)

#include<iostream>

using namespace std;

class dtMhs

{

private:

 struct data

 {

 int nim;

 char nama[30];

 float IPK;

 }x;

public:

 dtMhs()

 {

 x.nim=0;

 strcpy(x.nama," ");

 x.IPK=0;

 }

 void store(int ,char \*,float );

 void retrieve();

 ~dtMhs(){}

};

b.)

void dtMhs::store(int nim,char \*nama,float IPK)

{

 x.nim=nim;

 strcpy(x.nama,nama);

 x.IPK=IPK;

}

void dtMhs::retrieve()

{

 cout << "Nama\t: " << x.nama << endl << "NIM\t: " << x.nim << endl << "IPK\t: " << x.IPK << endl;

}

c.)

int main()

{

 dtMhs x;

 x.store(1701293931,"Henricko Randy",3.50);

 x.retrieve();

 cin.get();

}

Nama : Henricko Randy Setiawan

NIM : 1701293931

Kelas : 02PPT