Tugas 1: Implementasi Operasi Dasar Pada Array

Dwi Cahya Ramadani 5024201025

September 11, 2021

1 Listing Program

Berikut adalah Listing 1 yang merupakan jawaban dari ketiga pertanyaan pada tugas 1 kali ini. Pertanyaan nomor 1, yaitu fungsi untuk menghapus semua elemen array terjawab pada fungsi DeleteAll. Pertanyaan Nomor 2, yaitu untuk mencari nilai maksimum dari suatu array terjawab pada fungsi Biggest. Dan pertanyaan nomor 3, yaitu untuk menghapus elemen dengan nilai maksimum terjawab pada fungsi DeleteBiggest.

Listing 1: JAWABAN

```
#include <iostream>
#include <stdio.h>

void Cetak ( float L [100] , int M ){
    for(int i =0; i <= M ; i ++){
        printf ("%.2f\n",L [ i ]);
    }
}

void Insert ( float L [100] , int *M , int i , float X ){
    // Panjang array
    int N =100;
    int BACK ;
    if (* M <N -1){
        BACK =* M+1 ;
        // Menggeser Elemen Ke kanan
        while ( BACK > i ){
            L [ BACK ]= L [ BACK -1];
            BACK = BACK -1;
        }
    // Menyisipkan Nilai Pada posisi ke -i
    L [ i ]= X ;
    // Menambah Jumlah elemen
    *M = *M + 1;
    }
```

```
}

void Delete ( float L [100] , int *M , int i ){
    int BACK ;
    // Jika nilai i <0 atau i> posisi elemen maka tida tidak valid .
    if ((i <0)||( i >* M )) return ;
    BACK = i ;
    //Proses Menghapus dengan Menggeser ke Kiri
    for ( BACK = i ; BACK <* M ; BACK ++){
        L [ BACK ]= L [ BACK +1];
    }
    // Mengurangi posisi elemen terakhir
    *M = *M - 1;
}

int Search ( float L [100] , int M , float X ){
    // Parameter Input
    //L [100]: Array , panjang 100
    //M : Posisi Elemen terakhir
    //X : Nilai yang di cari
    // Output : Posisi elemen .
    // pos >=0 bila ditemukan
    // pos = -1 bila tidak ditemukan

    int C =0 , pos = -1;
    for ( C =0; C <= M ; C ++){
        if ( L [ C ]== X ){
            // Jika ditemukan maka
            // Simpan nilai C dan pencarian diakhiri .
            pos = C ;
            break ;
        }
    }
    return pos ;
}

void Update ( float L [100] , int i , float X ){
    // Parameter Input
    //L= Array panjang 10;
    //i: Posisi Elemen yang diperarui
    //X: Nilai yang baru

    L [ i ]= X ;
}

void Append ( float L [100] , int *M , float X ){
```

```
    // Definisikan Panjang array
    int N =100;
    if (*M <N -1){
        // Menambah Jumlah Elemen
        *M = *M + 1;
        // Memasukkan X sebagai nilai index terakhir
        L [*M ]= X ;
    }
}

void DeleteAll (float L[100], int *M){
    //Definisikan Akhir Array
    int BACK = *M;
    //Proses penghapusan
    for (int i=BACK ; i>=0; i--){
        Delete(L,&BACK,i);
    }
    *M = BACK;
}

float Biggest (float L[100], int M, int *pos){
    float temp=0;
    // Traversal untuk mencari nilai terbesar
    for (int i=0; i<=M; i++){
        if (L[i]>temp){
            temp=L[i];
            *pos=i;
        }
    }
    return temp;
}

void DeleteBiggest (float L[100], int *M, float big){
    //Mendefinisikan Akhir Array
    int BACK = *M;
    //Mencari elemen yang memiliki nilai sama dengan nilai maksimum
    for (int i=BACK; i>=0; i--){
        if (L[i]==big){
            //Proses Menghapus
            Delete(L,&BACK,i);
        }
    }
    *M = BACK;
}

int main (){
```

```
// Index array pada C dimulai dari 0.
float L [100];

//M= -1 Array kosong
int M;
M = -1;

// Menambah lima Item pada array .
printf("Item Telah Ditambahkan!\n");
Append (L ,& M ,4);
Append (L ,& M ,5);
Append (L ,& M ,6);
Append (L ,& M ,5);
Append (L ,& M ,1);
Cetak (L , M );
system("PAUSE");

// Mencari Posisi item yang bernilai 6.
int pos ;
int x=6;
printf("\nMencari item bernilai %d...\n",x);
pos = Search (L ,M ,x);
printf ("Posisi item di index ke-%d !\n", pos );
system("PAUSE");


// Hapus elemen yang telah disimpan di variabel pos .
printf("\nMenghapus item di index ke %d...\n", pos);
Delete (L ,& M , pos );
printf("Item Telah Dihapus !\n");
Cetak (L , M );
system("PAUSE");

// Menyisipkan nilai 8 pada posisi index 4;
float a=15;
int b=2;
printf("\nMenyisipkan nilai %.f pada index ke %d...\n",a,b);
Insert (L ,& M ,b, a);
printf("Item Telah Ditambahkan !\n");
Cetak (L , M );
system("PAUSE");

//Menghapus Semua Nilai dari Array
printf("\nMenghapus semua item...\n");
DeleteAll(L, &M);
printf("Item Telah Dihapus !\n");
```

```
    system("PAUSE");

    //Menambahkan Item Baru
    system("CLS");
    printf("Membuat Array Baru...\n");
    Append (L ,& M ,15);
    Append (L ,& M ,4);
    Append (L ,& M ,10);
    Append (L ,& M ,1);
    Append (L ,& M ,10);
    Append (L ,& M ,9);
    printf("Array Baru Telah Dibuat !\n");
    Cetak (L , M );
    system("PAUSE");

    //Mengubah Nilai
    printf("\nMengubah nilai item index ke-%d menjadi %.f\n",b,a);
    Update(L, b, a);
    printf("Nilai telah diubah !\n");
    Cetak (L, M);
    system("PAUSE");

    //Mencari Nilai Maksimum
    printf("\nMencari nilai maksimum...\n");
    float big;
    big=Biggest(L,M,&pos);
    printf("Nilai maksimumnya adalah %.f pada posisi %d\n",big,pos);
    system("PAUSE");

    //Menghapus Nilai Maksimum
    printf("\nMenghapus nilai maksimum...\n");
    DeleteBiggest(L,&M,big);
    printf("Data Dengan Nilai %.f Telah Dihapus!\n",big);
    Cetak (L , M );

    return 0;
}
```

2 Penjelasan

Pada program yang telah dibuat, saya menambahkan fungsi DeleteAll untuk menghapus seluruh elemen array. Dalam fungsi tersebut terdapat variabel BACK yang saya gunakan untuk menandai akhir array, lalu dari akhir array tersebut saya lakukan perulangan untuk operasi Delete hingga ke elemen paling awal. Setiap melakukan operasi Delete nilai variabel BACK akan dikurangi 1, dan pada akhirnya variabel M sebagai penghitung jumlah elemen akan diisi dengan nilai BACK yang telah menjadi -1.

Fungsi selanjutnya yang saya tambahkan adalah fungsi Biggest dan DeleteBiggest. Dalam fungsi Biggest saya melakukan operasi traversal atau mengunjungi elemen 1 persatu. Setiap mengunjungi suatu elemen, akan terjadi perbandingan nilai elemen tersebut dengan nilai variabel temp yang saya set nilai awalnya 0. Jika nilai elemen lebih besar dari nilai temp, maka nilai temp akan diubah menjadi nilai elemen tersebut dan . Pada akhirnya fungsi tersebut akan mengembalikan nilai temp yang akan diterima oleh variabel big pada fungsi main. Sedangkan pada fungsi DeleteBiggest, pertama-tama saya definisikan variabel BACK sebagai akhir array, lalu terjadi proses traversal mulai dari elemen terakhir hingga elemen pertama.Dalam setiap kunjungan akan dilakukan perbandingan nilai elemen tersebut dengan nilai variabel big. Ketika elemen tersebut bernilai sama dengan big,maka akan terjadi fungsi Delete yang akan menghapus elemen tersebut sekaligus mengurangi 1 dari nilai variabel BACK yang menandakan jumlah elemen berkurang 1. Pada akhirnya variabel BACK akan diisikan pada variabel M sebagai jumlah elemen terbaru.

Meski begitu, masih terdapat kesalahan dalam program tersebut. Jika ada lebih dari 1 elemen yang memiliki nilai maksimum, maka fungsi Biggest hanya akan mengembalikan posisi elemen maksimum yang pertama. Yang mana seharusnya semua posisi ditampilkan. Namun untuk fungsi DeleteBiggest sudah bisa menghapus semua elemen yang bernilai maksimum.