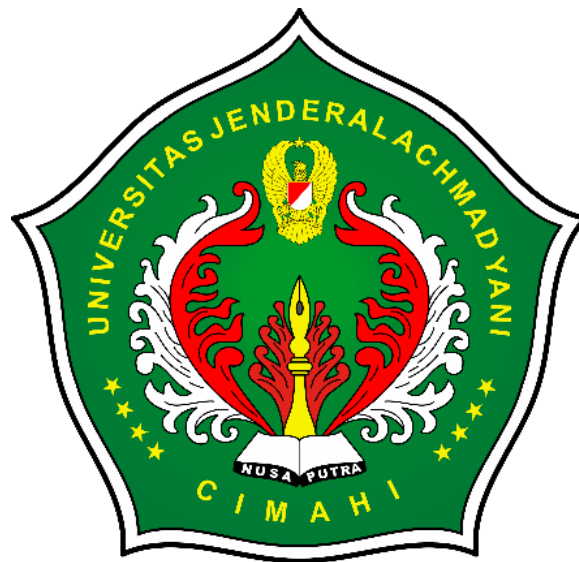


**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL 3  
ADT TabInt (Tabel Integer)**

**DISUSUN OLEH :  
AJI KARTIKO HARTANTO - 2350081062**



**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN INFORMATIKA  
UNIVERSITAS JENDERAL ACHMAD YANI  
TAHUN 2023**

# DAFTAR ISI

DAFTAR GAMBAR .....	ii
BAB I. HASIL PRAKTIKUM.....	1
I.1 Program garis.c.....	1
I.1.A. Boolean.h .....	<b>Kesalahan! Bookmark tidak ditentukan.</b>
I.1.B. Source Code garis.c .....	1
I.1.C. Hasil .....	7
I.1.D. Analisa .....	8
BAB II. TUGAS PRAKTIKUM .....	9
II.1 Tugas ADT Line.....	9
II.1.A. Source Code Boolean.h .....	9
II.1.B. Source Code line.c.....	9
II.1.C. Source Code line.h .....	11
II.1.D. Source Code mline.c .....	16
II.1.E. Source Code point.c .....	<b>Kesalahan! Bookmark tidak ditentukan.</b>
II.1.F. Source Code point.h.....	<b>Kesalahan! Bookmark tidak ditentukan.</b>
II.1.G. Hasil .....	19
II.1.H. Analisa.....	20
BAB III. KESIMPULAN .....	21

## DAFTAR GAMBAR

Gambar I.1 Output Program garis.c .....	7
Gambar II.1 Output Program Tugas ADT TabInt (mtabint.c/main driver) .....	19

# BAB I. HASIL PRAKTIKUM

## I.1 Program tabpoint Latihan

### I.1.A. Source Code tabpoint.c

```
/*
    Program          : TabPoint.c
    Author           : 2350081062, Aji Kartiko Hartanto
    Kelas            : C
    Deskripsi         :
    Tanggal           : 27-03-2024
*/

#include <stdio.h>
#include <conio.h>

//definisi nilai max = 10
#define nMax 10

typedef struct{
    int abs;
    int ord;
}Point;

typedef struct{
    Point Tp[nMax + 1];
    int nEff;
}TabPoint;

//prototype
void CreateTabPoint(TabPoint *T);
void AddElmTab(TabPoint *T, Point P);
void CetakTab(TabPoint T);

//tugas
```

```

void AddFirstElm(TabPoint *T, Point P);
void CetakInversTab(TabPoint T);
void CetakKuadranPoint(TabPoint T);
TabPoint CopyTabNeg(TabPoint T);
int PointMax(TabPoint T);
int kuadran(Point P);
void InversTab(TabPoint *T);

//main driver
int main(){
    //kamus
    TabPoint MyTab;
    int x, y, x1, y1;
    Point P;

    //algoritma
    CreateTabPoint(&MyTab);
    printf("input x: "); scanf("%d", &x);
    printf("input y: "); scanf("%d", &y);

    while( x != 999 && y != 999 ){
        P.abs = x;
        P.ord = y;

        AddElmTab(&MyTab, P);
        printf("input x: "); scanf("%d", &x);
        printf("input y: "); scanf("%d", &y);
    }
    CetakTab(MyTab);

    printf("\n");
    printf("input x baru: "); scanf("%d", &x1);
    printf("input y baru: "); scanf("%d", &y1);
    P.abs = x1;
    P.ord = y1;
    AddFirstElm(&MyTab, P);

```

```

        CetakTab(MyTab);

        printf("\n");
        printf("Tab dalam reverse order:\n");
        CetakInversTab(MyTab);

        printf("\n");
        printf("Kuadran of each point:\n");
        CetakKuadranPoint(MyTab);

        printf("\n");
        printf("Copying points dengan negasi:\n");
        TabPoint negTab = CopyTabNeg(MyTab);
        CetakTab(negTab);

        printf("\n");
        int maxIndex = PointMax(MyTab);
        printf("Index point dengan maksimum jaran from origin: %d\n",
maxIndex);

        printf("\n");
        printf("Inversi Tab nya:\n");
        InversTab(&MyTab);
        CetakTab(MyTab);

        return 0;
}

//realisasi prototype
void CreateTabPoint(TabPoint *T){
    (*T).nEff = 0;
}

void AddElmTab(TabPoint *T, Point P){
    if((*T).nEff < nMax){
        (*T).nEff++;
    }
}

```

```

        (*T).Tp[(*T).nEff] = P;

    }

}

void CetakTab(TabPoint T){
    //kamus
    int i, x, y;
    //algoritma
    for(i = 1; i <= T.nEff; i++){
        x = T.Tp[i].abs;
        y = T.Tp[i].ord;
        printf("< %d , %d >", x, y);

    }

}

//realisasi prototype tugas
void AddFirstElm(TabPoint *T, Point P){
    if((*T).nEff < nMax){
        int i;
        for(i = (*T).nEff; i > 0; i--){
            (*T).Tp[i + 1] = (*T).Tp[i];
        }
        (*T).Tp[1] = P;
        (*T).nEff++;
    }

}

void CetakInversTab(TabPoint T){
    int i;
    for(i = T.nEff; i > 0; i--){
        printf("< %d , %d >", T.Tp[i].abs, T.Tp[i].ord);
    }

}

void CetakKuadranPoint(TabPoint T){
    int i;
    for(i = 1; i <= T.nEff; i++){

```

```

        printf("Point < %d , %d > adalah di kuadran %d\n",
T.Tp[i].abs, T.Tp[i].ord, kuadran(T.Tp[i]));
    }
}

TabPoint CopyTabNeg(TabPoint T){
    TabPoint result;
    CreateTabPoint(&result);
    int i;

    for(i = 1; i <= T.nEff; i++){
        if(T.Tp[i].ord < 0){
            AddElmTab(&result, T.Tp[i]);
        }
    }

    return result;
}

int PointMax(TabPoint T){
    int maxIndex = 1;
    double maxDistance = T.Tp[1].abs * T.Tp[1].abs + T.Tp[1].ord *
T.Tp[1].ord;
    int i;

    for(i = 2; i <= T.nEff; i++){
        double distance = T.Tp[i].abs * T.Tp[i].abs + T.Tp[i].ord *
T.Tp[i].ord;
        if(distance > maxDistance){
            maxDistance = distance;
            maxIndex = i;
        }
    }

    return maxIndex;
}

```



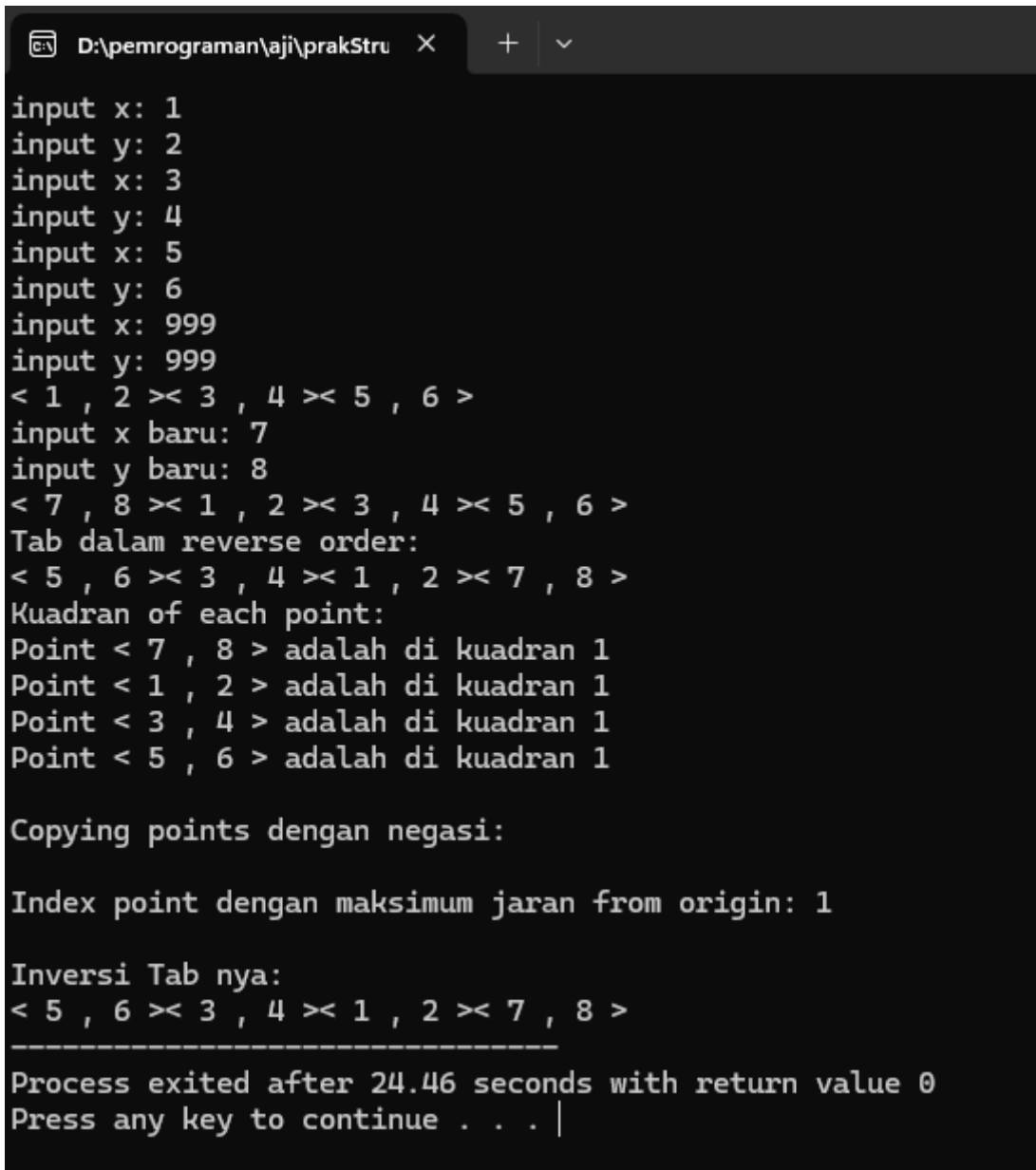
```

int kuadran(Point P){
    if(P.abs > 0 && P.ord > 0)
        return 1;
    else if(P.abs < 0 && P.ord > 0)
        return 2;
    else if(P.abs < 0 && P.ord < 0)
        return 3;
    else if(P.abs > 0 && P.ord < 0)
        return 4;
    else
        return 0;
}

void InversTab(TabPoint *T){
    Point temp;
    int i;
    for(i = 1; i <= T->nEff / 2; i++){
        temp = T->Tp[i];
        T->Tp[i] = T->Tp[T->nEff - i + 1];
        T->Tp[T->nEff - i + 1] = temp;
    }
}

```

### I.1.B. Hasil



```
D:\pemrograman\aji\prakStru X + v
input x: 1
input y: 2
input x: 3
input y: 4
input x: 5
input y: 6
input x: 999
input y: 999
< 1 , 2 >< 3 , 4 >< 5 , 6 >
input x baru: 7
input y baru: 8
< 7 , 8 >< 1 , 2 >< 3 , 4 >< 5 , 6 >
Tab dalam reverse order:
< 5 , 6 >< 3 , 4 >< 1 , 2 >< 7 , 8 >
Kuadran of each point:
Point < 7 , 8 > adalah di kuadran 1
Point < 1 , 2 > adalah di kuadran 1
Point < 3 , 4 > adalah di kuadran 1
Point < 5 , 6 > adalah di kuadran 1

Copying points dengan negasi:

Index point dengan maksimum jaran from origin: 1

Inversi Tab nya:
< 5 , 6 >< 3 , 4 >< 1 , 2 >< 7 , 8 >
-----
Process exited after 24.46 seconds with return value 0
Press any key to continue . . . |
```

Gambar 1.1 Output Program garis.c

### **I.1.C. Analisa**

Program ini adalah program yang dibuat dengan Bahasa C. Program ini adalah implementasi bahasa C yang dirancang untuk menangani data titik dua dimensi. Ini menggunakan struktur data Point untuk menampilkan titik dalam ruang dua dimensi dan TabPoint untuk menyimpan kumpulan titik dalam bentuk array. Dengan fungsi-fungsinya yang telah didefinisikan, program ini memungkinkan pengguna melakukan berbagai operasi, seperti menambahkan titik baru ke kumpulan titik, mencetak kumpulan titik dalam berbagai format, atau mencetak kumpulan titik dalam berbagai format.

Dalam fungsi utama (main), program meminta pengguna untuk memasukkan koordinat titik secara berurutan, yang kemudian dimasukkan ke dalam kumpulan titik. Setelah kumpulan titik terbentuk, program dapat melakukan berbagai operasi, seperti menambahkan titik baru ke posisi pertama dalam kumpulan titik, mencetak kumpulan titik dalam urutan terbalik, menentukan kuadran dari setiap titik, dan membuat salinan kumpulan titik yang hanya mengandung titik-titik dengan ordinat ne Akhirnya, program mencetak hasil operasi sehingga pengguna dapat melihatnya. Contoh penggunaan struktur data dan fungsi bahasa C untuk memanipulasi data geometris tersedia dalam program ini.

## BAB II. TUGAS PRAKTIKUM

### II.1 Tugas ADT TabInt (Tabel Integer)

#### II.1.A. Source Code Boolean.h

```
/*
    Program      : boolean.h
    Deskripsi    : Header dari file boolean
*/
#ifndef boolean_H
#define true 1
#define false 0
#define boolean unsigned char
#endif
```

#### II.1.B. Source Code tabint.h

```
/*
    Program      : tabint.h
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas       : C
    Deskripsi    : Header file dari prototype Tab Integer
    Tanggal     : 27-03-2024
*/

#ifndef _TABINT_H
#define _TABINT_H
#include "boolean.h"
#include <stdio.h>
#include <conio.h>
#define nMax 10
#define ElType int
#define IdxUndef -999
typedef struct
{
    ElType T1[nMax +1];
    int neff;
```

```

    /* data */
}TabInt;

/*konstruktor*/
void CreateTabInt(TabInt *T);

/*selektor tabint*/
int GetJumElmt(TabInt T);
int GetFirstIdx(TabInt T);
int GetLlastIdx(TabInt T);
int GetElmt(TabInt T, int index);

/*set nilai*/
void SetElm(TabInt *T,int i,int v);

/*kelompok operasi cek elemen kosong atau penuh*/
boolean isEmpty(TabInt T);
boolean IsFull(TabInt T);

/*kelompok operasi input output device*/
void BacaElm(TabInt *T);
void CetakTabInt(TabInt T);
void AddElm(TabInt *T,int x);

/*kelompok operasi aritmatika*/
TabInt KaliTab(TabInt Tab1,TabInt Tab2);
TabInt KaliKons(TabInt T, int c);

/*kelompok operasi relasional terhadap TabInt*/
boolean IsEQTab(TabInt Tab1, TabInt Tab2);

```

```

/*kelompok operasi lain terhadap type*/
void CopyTab(TabInt Tabin, TabInt *Tabout);
TabInt InversTab(TabInt T);
boolean IsElemenAda(TabInt T, ElType X);
int CariElemen(TabInt T, ElType X);

#endif

```

### II.1.C. Source Code tabint.c

```

/*
    Program      : tabint.c
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas        : C
    Deskripsi    : Memasukan instruksi yang akan dieksekusi didalam
fungsi dan prosedur
    Tanggal      : 28-03-2024
*/

#include <stdio.h>
#include <conio.h>
#include "tabint.h"

/*konstruktor*/
void CreateTabInt(TabInt *T)
{
    (*T).neff = 0;
}

/*selektor tabint*/
int GetJumElmt(TabInt T)
{
    return T.neff;
}

int GetFirstIdx(TabInt T)
{
    return T.T1[1];
}

```

```

}
int GetLlastIdx(TabInt T)
{
    return T.Tl[T.neff];
}
int GetElmt(TabInt T, int index)
{
    return T.Tl[index];
}
/*set nilai*/
void SetElm(TabInt *T, int i, int v)
{
    (*T).Tl[i] = v;
}
/*kelompok operasi cek elemen kosong atau penuh*/
boolean isEmpty(TabInt T)
{
    if (T.neff == 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}
boolean IsFull(TabInt T)
{
    if (T.neff == nMax)
    {
        return true;
    }
    else
    {
        return false;
    }
}

```

```

/*kelompok operasi input output device*/
void BacaElm(TabInt *T)
{
    int Elm;
    printf("Masukan Elemen :");
    scanf("%d", &Elm);
    while (Elm != 999)
    {
        AddElm(&(*T), Elm);
        printf("Masukan Elemen :");
        scanf("%d", &Elm);
    }
}

void CetakTabInt(TabInt T)
{
    for (int i = 1; i <= T.neff; i++)
    {
        printf("[%d]", T.T1[i]);
    }
}

void AddElm(TabInt *T, int x)
{
    if (!IsFull((*T)))
    {
        (*T).neff++;
        (*T).T1[(*T).neff] = x;
    }
}

/*kelompok operasi aritmatika*/
TabInt KaliTab(TabInt Tab1, TabInt Tab2)
{
    TabInt NewTAB;

    CreateTabInt(&NewTAB);
    if (!isEmpty(Tab1) && !isEmpty(Tab2) && Tab1.neff == Tab2.neff)
    {
        for (int i = 1; i <= Tab1.neff; i++)

```



```

        {
            AddElm(&NewTab, Tab1.T1[i] * Tab2.T1[i]);
        }
    }
    return NewTab;
}

TabInt KaliKons(TabInt T, int c)
{
    TabInt NewTab;
    CreateTabInt(&NewTab);
    for (int i = 1; i <= T.neff; i++)
    {
        AddElm(&NewTab, T.T1[i] * c);
    }
    return NewTab;
}

/*kelompok operasi relasional terhadap TabInt*/
boolean IsEQTab(TabInt Tab1, TabInt Tab2)
{
    if (Tab1.neff == Tab2.neff)
    {
        return true;
    }
    else
    {
        return false;
    }
}

/*kelompok operasi lain terhadap type*/
void CopyTab(TabInt Tabin, TabInt *Tabout)
{
    (*Tabout) = Tabin;
}

TabInt InversTab(TabInt T)
{
    TabInt NewTab;
    CreateTabInt(&NewTab);

```

```

    for (int i = T.neff; i >= 1; i--)
    {
        AddElm(&NewTab, T.T1[i]);
    }

    return NewTab;
}

boolean IsElemenAda(TabInt T, ElType X)
{
    int i;
    boolean hasil;
    i = 1;
    hasil = false;
    while (i <= T.neff && !hasil)
    {
        if (T.T1[i] == X)
        {
            hasil = true;
        }
        else
        {
            i++;
        }
    }

    return hasil;
}

int CariElemen(TabInt T, ElType X)
{
    int i = 1;
    boolean ketemu;
    ketemu = false;
    while (i <= T.neff && !ketemu)
    {
        if (T.T1[i] == X)
        {
            return i;
        }
    }
}

```

```

        ketemu = true;
    }
    else
    {
        i++;
    }
}

return IdxUndef;
}

```

#### II.1.D. Source Code mtabint.c

```

/*
    Program      : tabint.c
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas        : C
    Deskripsi    : Main Driver dari 3 program yang telah dibuat
    Tanggal      : 30-03-2024
*/

#include <stdio.h>
#include <stdlib.h>
#include "tabint.h"
#include "tabint.c"

int main()
{
    TabInt MyTab, Tab1, Tab2, Tab3, TabCopy;
    int index, newElm, cons, Elm;

    CreateTabInt (&MyTab);
    BacaElm (&MyTab);
    CetakTabInt (MyTab);

    /*selektor*/

```

```

printf("\nJumlah elemen pada tabel : %d", GetJumElmt(MyTab));
printf("\nElemen pertama pada tabel : %d", GetFirstIdx(MyTab));
printf("\nElemen terakhir pada tabel : %d", GetLlastIdx(MyTab));
printf("\nMasukan index elemen yang akan di ambil : ");
scanf("%d", &index);
printf("\nElemen index ke-%d adalah : %d", index, GetElmt(MyTab,
index));

printf("\nMasukan index elemen yang akan di ubah :");
scanf("%d", &index);
printf("\nMasukan elemen yang baru :");
scanf("%d", &newElm);
SetElm(&MyTab, index, newElm);
CetakTabInt(MyTab);
if (isEmpty(MyTab))
{
    printf("\nTabel kosong");
    /* code */
}
else
{
    printf("\nTabel tidak kosong");
}
if (IsFull(MyTab))
{
    printf("\nTabel penuh");
    /* code */
}
else
{
    printf("\nTabel tidak penuh");
}
printf("\n\nMasukan elemen Tab1 untuk perkalian\n");
CreateTabInt(&Tab1);
BacaElm(&Tab1);
CetakTabInt(Tab1);
CreateTabInt(&Tab2);
Tab2 = KaliTab(MyTab, Tab1);

```

```

printf("\nHasil kali antar tabel Tab2:");
CetakTabInt(Tab2);
printf("\nMasukan konstasta yang akan di kalikan :");
scanf("%d", &cons);
CreateTabInt(&Tab3);
Tab3 = KaliKons(MyTab, cons);
printf("\nHasil perkalian tabel dengan konstasta Tab3:");
CetakTabInt(Tab3);
if (IsEQTab(MyTab, Tab1))
{
    printf("\n\n MyTab dan Tab1 memiliki jumlah elemen yang
sama");
    /* code */
}
else
{
    printf("\n\n MyTab dan Tab1 memiliki jumlah elemen
berbeda");
}
CreateTabInt(&TabCopy);
CopyTab(MyTab, &TabCopy);
printf("\nOperasi menyalin tabel MyTab ke TabCopy :");
CetakTabInt(TabCopy);
printf("\nInvers MyTab : ");
CetakTabInt(InversTab(TabCopy));
printf("\nMasukan elemen yang akan di cari pada TabCopy :");
scanf("%d", &Elm);
if (IsElemenAda(TabCopy, Elm))
{
    printf("Elemen Ada");
    /* code */
}
else
{
    printf("Elemen Tidak ada");
}
printf("\nMasukan elemen yang akan di cari pada MyTab :");

```

```

scanf("%d", &Elm);

printf("\nElemen yang di cari berada pada index ke-%d",
CariElemen(MyTab, Elm));

return 0;
}

```

### II.1.E. Hasil

```

PS D:\pemrograman\aji\prakStrukDat\pert3\ADT TabInt> cd "d:\pemrograman\aji\prakStrukDat\pert3\ADT TabInt\" ; if ($?) { gcc mtabint.c -o mtabint
} ; if ($?) { .\mtabint }
Masukan Elemen :1
Masukan Elemen :2
Masukan Elemen :3
Masukan Elemen :4
Masukan Elemen :5
Masukan Elemen :999
[1][2][3][4][5]
Jumlah elemen pada tabel : 5
Elemen pertama pada tabel : 1
Elemen terakhir pada tabel : 5
Masukan index elemen yang akan di ambil : 4

Elemen index ke-4 adalah : 4
Masukan index elemen yang akan di ubah :3

Masukan elemen yang baru :80
[1][2][80][4][5]
Tabel tidak kosong
Tabel tidak penuh

Masukan elemen Tab1 untuk perkalian
Masukan Elemen :1
Masukan Elemen :10
Masukan Elemen :2
Masukan Elemen :3
Masukan Elemen :4
Masukan Elemen :999
[1][10][2][3][4]
Hasil kali antar tabel Tab2:[1][20][160][12][20]
Masukan konstasta yang akan di kalikan :12

Hasil perkalian tabel dengan konstasta Tab3:[12][24][960][48][60]

MyTab dan Tab1 memiliki jumlah elemen yang sama

```

*Gambar II.1 Output Program Tugas ADT TabInt (mtabint.c/main driver)*

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Masukan Elemen :2
Masukan Elemen :3
Masukan Elemen :4
Masukan Elemen :999
[1][10][2][3][4]
Hasil kali antar tabel Tab2:[1][20][160][12][20]
Masukan konstasta yang akan di kalikan :12

Hasil perkalian tabel dengan konstasta Tab3:[12][24][960][48][60]

MyTab dan Tab1 memiliki jumlah elemen yang sama
Operasi menyalin tabel MyTab ke TabCopy :[1][2][80][4][5]
Invers MyTab : [5][4][80][2][1]
Masukan elemen yang akan di cari pada TabCopy :2
Elemen Ada
Masukan elemen yang akan di cari pada MyTab :5

Elemen yang di cari berada pada index ke-5
PS D:\pemrograman\aji\prakStrukDat\pert3\ADT TabInt> 
```

*Gambar 1 Output Program Tugas ADT TabInt (mtabint.c/main driver)(1)*

#### **II.1.F. Analisa**

Program ini menggunakan struktur data array (dikenal sebagai TabInt) untuk membuat array, membaca elemennya, mencetak isi array, dan melakukan manipulasi elemen, seperti mengambil dan mengubah nilai pada indeks tertentu. Program juga melakukan operasi tambahan, seperti perkalian antara array dan perkalian dengan konstanta, dan pengecekan apakah dua array memiliki kesamaan jumlah elemen.

Program ini menggunakan alokasi statis untuk array, yang dapat dianggap cukup efisien dalam penggunaan memori. Namun, masih ada ruang untuk meningkatkan efisiensi program dengan mengoptimalkan operasi-operasi yang digunakan dan penggunaan memori. Program ini juga memiliki antarmuka pengguna yang mudah digunakan, yang memungkinkan pengguna dengan mudah berinteraksi dengan program melalui konsol.

Namun, masalah keamanan program ini termasuk kesalahan pengguna seperti memasukkan indeks di luar rentang array atau melakukan operasi pada array yang belum diinisialisasi. Untuk meningkatkan keamanan, program dapat diperbaiki dengan menambahkan validasi input dan menangani masalah khusus yang mungkin terjadi selama eksekusi program. Dengan demikian, program ini dapat menjadi alat yang lebih aman dan bermanfaat untuk mengelola array.

### **BAB III. KESIMPULAN**

Pada pertemuan modul 3 ini saya telah belajar mengenai bab ADT Tabel Integer. Sedikit menjelaskan mengenai bagian bab tugas praktikum, yaitu Tujuan program ini adalah untuk memanipulasi struktur data array bahasa C yang dikenal sebagai TabInt. Program ini dapat melakukan beberapa fungsi dasar seperti membuat array, membaca elemen, dan mencetak array, serta operasi tambahan seperti mengambil elemen pada indeks tertentu, mengubah nilai elemen pada indeks tertentu, dan mengecek apakah array kosong atau penuh.

Selain itu, program ini dapat melakukan operasi tambahan seperti perkalian antara array dengan konstanta dan antara array dengan lainnya, menguji apakah dua array memiliki jumlah elemen yang sama, menyalin isi array ke array lain, mencari elemen dalam array, dan melakukan inversi terhadap isi array. Dengan demikian, program ini memberikan contoh penggunaan struktur data array dan implementasi fungsi-fungsi untuk melakukan berbagai operasi yang umum dilakukan pada array dalam bahasa pemrograman C.