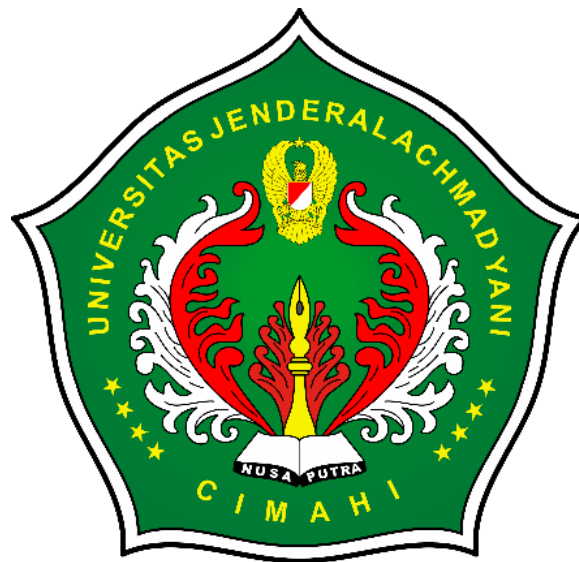


**LAPORAN PRAKTIKUM
STRUKTUR DATA**

**MODUL 1
PERKENALAN ADT**

**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 ADT Jam	1
I.1.A. Boolean.h	1
I.1.B. Source Code jam.h	1
I.1.C. Source Code jam.c	5
I.1.D. Source Code mjam.c	10
I.1.E. Hasil	12
I.1.F. Analisa	13
BAB II. TUGAS PRAKTIKUM	14
II.1 Tugas ADT Point	14
II.1.A. Source Code Boolean.h	14
II.1.B. Source Code Point.h	14
II.1.C. Source Code Point.c	17
II.1.D. Source Code mpoint.c	23
II.1.E. Hasil	26
II.1.F. Analisa	26
BAB III. KESIMPULAN	28

DAFTAR GAMBAR

Gambar I.1 Output Program ADT Jam.....	12
Gambar II.1 Output Program Tugas ADT Point.....	26

BAB I. HASIL PRAKTIKUM

I.1 Program ADT Jam

I.1.A. Boolean.h

```
/*
    Program      : boolean.h
    Deskripsi    : header file boolean
*/

#ifndef boolean_H
#define boolean_H
#define true 1
#define false 0
#define boolean unsigned char
#endif
```

I.1.B. Source Code jam.h

```
/*
    Program      : jam.h
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas       : C
    Deskripsi    : Header file dari prototype jam
    tanggal     : 15-03-2024
*/

#ifndef JAM_H
#define JAM_H
#include "boolean.h"
#define true 1
#define false 0
#define boolean unsigned char

typedef struct {
int Hour ; /* Hour [0..23] */
int Minute; /* Minute [0..59] */
int Second; /* Second [0..59] */
} JAM;
```

```

/* Prototype ADT Jam */

/** Konstruktor membentuk ADT Jam **/
void CreateJam (JAM *J, int HH, int MM, int SS);
/*
    Membentuk sebuah JAM dari komponen-komponennya yang valid
    Pre condition : HH,MM,SS valid untuk membentuk JAM
    I.S : J tidak terdefinisi, tidak diketahui nilainya
    F.S : membentuk sebuah Jam dari komponen-komponennya dengan J
    diinisialisasi nilainya dengan Hour = HH, Minute = MM, Second = SS
*/

/* Selektor nilai JAM **/
int GetHour(JAM J);
// Mengirimkan komponen Hour dari J
int GetMinute(JAM J);
// Mengirimkan komponen Minute dari J
int GetSecond(JAM J);
// Mengirimkan komponen Second dari J

/* Set nilai komponen */
void SetHour(JAM *J, int newHour);
/* I.S : J terdefinisi, dengan nilainya sembarang
    F.S : Mengubah nilai komponen Hour dari J dengan Hour=newHour
*/
void SetMinute(JAM *J, int newMinute);
/* I.S : J terdefinisi, dengan nilainya sembarang
    F.S : Mengubah nilai komponen MMinute dari J dengan Minute=newMinute
*/
void SetSecond(JAM *J, int newSecond);
/* I.S : J terdefinisi, dengan nilainya sembarang
    F.S : Mengubah nilai komponen Second dari J dengan Second=newSecond
*/

/* Destruktor/Dealokator: tidak perlu */

/** {Kelompok Baca/Tulis} **/
void ReadJam (JAM *J);
/* I.S. : J tidak terdefinisi
    F.S. : J terdefinisi dan merupakan jam yang valid

```

Proses : proses membaca komponen H,M,S terus berlangsung sampai membentuk J yang valid. Komponen jam dikatakan valid, Jika $0 \leq H \leq 23$, $0 \leq M \leq 60$, dan $0 \leq S \leq 59$

```
*/
void PrintJam (JAM J);
/* I.S. : J diisi dengan nilai sembarang
   F.S. : Menampilkan jam (J) ke layar dengan format HH:MM:SS
*/

/** { kelompok Validasi Type } **/
boolean IsJValid (int H, int M, int S);
/* fungsi pengecekan nilai komponen jam yang valid
   Mengirimkan true jika H,M,S dapat membentuk J yang valid
   Sebaliknya mengirimkan false jika H,M,S tidak dapat membentuk J yang
valid
*/

/** {Operator Relasional} **/
boolean JEQ(JAM J1, JAM J2);
/* Mengirimkan true jika J1=J2, false jika tidak
   Artinya mengirimkan true jika semua elemen jam bernilai bernilai sama
   Contoh :
- Jika J1 = 10:20:30 dan J2 = 10:20:30, maka mengirimkan true
- Jika J1 = 10:20:30 dan J2 = 10:30:30, maka mengirimkan false
*/
boolean JLT(JAM J1, JAM J2);
/* Mengirimkan true jika J1<J2 , false jika tidak
   Artinya mengirimkan true jika elemen pada J1 bernilai kurang dari J2
   Contoh :
- Jika J1 = 10:20:30 dan J2 = 10:30:30, maka mengirimkan true
- Jika J1 = 10:30:30 dan J2 = 10:30:30, maka mengirimkan false
- Jika J1 = 11:30:30 dan J2 = 10:30:30, maka mengirimkan false
*/
boolean JGT(JAM J1, JAM J2);
/* Mengirimkan true jika J1>J2, false jika tidak
   Artinya mengirimkan true jika elemen pada J1 bernilai lebih dari J2
   Contoh :
- Jika J1 = 10:40:30 dan J2 = 10:30:30, maka mengirimkan true
- Jika J1 = 10:30:30 dan J2 = 10:30:30, maka mengirimkan false
- Jika J1 = 10:20:30 dan J2 = 11:20:30, maka mengirimkan false
*/
```

```

/** { operator aritmatika } **/
void Reset (JAM *J);
/* I.S. : J sembarang
   F.S. : J bernilai "nol", yaitu semua komponen bernilai 0
   Proses : Ubah semua nilai elemen J menjadi 0:0:0
*/
JAM NextDetik (JAM J);
/* Menambahkan nilai 1 detik pada jam, dan kirimkan nilai jam baru */
JAM NextNDetik (JAM J, int N);
/* Menambahkan nilai N detik pada jam, dan mengirimkan nilai jam baru */
long int Durasi ( JAM Jaw , JAM JAkh);
/* Menghitung selisih dari dua buah jam dalam bentuk detik
   Proses : Mengirimkan JAkh -JAw dlm Detik, Hasilnya negatif jika Jaw >
JAkhir
*/

/** {Kelompok Konversi Terhadap Type} **/
long int JamToDetik (JAM J);
/* melakukan konversi jam menjadi Detik
   Rumus : detik = 3600*hour+menit*60 + detik
   nilai maksimum = 3600*23+59*60+59*60
*/

#endif

```

I.1.C. Source Code jam.c

```
/*
    Program      : jam.c
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas        : C
    Deskripsi     : Body file dari prototype jam
    tanggal      : 15-03-2024
*/

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

/* Realisasi dari prototype ADT Jam */
/** Konstruktor membentuk ADT Jam **/
void CreateJam (JAM *J, int HH, int MM, int SS) {
    /* Membentuk sebuah JAM dari komponen-komponennya yang valid
       Pre condition : HH,MM,SS valid untuk membentuk JAM
       I.S : J tidak terdefinisi, tidak diketahui nilainya
       F.S : membentuk sebuah Jam dari komponen-komponennya dengan J
       diinisialisasi nilainya dengan Hour = HH, Minute = MM, Second = SS
    */
    (*J).Hour = HH;
    (*J).Minute = MM;
    (*J).Second = SS;
}

/* Selektor nilai JAM */
int GetHour(JAM J) {
    // Mengirimkan komponen Hour dari J
    return (J.Hour);
}
int GetMinute(JAM J){
    // Mengirimkan komponen Minute dari J
    return (J.Minute);
}
int GetSecond(JAM J) {
    // Mengirimkan komponen Second dari J
    return (J.Second);
}
```



```

/* Set nilai komponen */
void SetHour(JAM *J, int newHour) {
/* I.S : J terdefinisi, dengan nilainya sembarang
   F.S : Mengubah nilai komponen Hour dari J dengan Hour=newHour
*/
(*J).Hour = newHour;
}

void SetMinute(JAM *J, int newMinute) {
/* I.S : J terdefinisi, dengan nilainya sembarang
   F.S : Mengubah nilai komponen Minute dari J dengan Minute=newMinute
*/
(*J).Minute = newMinute;
}

void SetSecond(JAM *J, int newSecond) {
/* I.S : J terdefinisi, dengan nilainya sembarang
   F.S : Mengubah nilai komponen Second dari J dengan Second=newSecond
*/
(*J).Second = newSecond;
}

/** {Kelompok Baca/Tulis} **/
void ReadJam (JAM *J) {
/* I.S. : J tidak terdefinisi
   F.S. : J terdefinisi dan merupakan jam yang valid
   Proses : proses membaca komponen H,M,S terus berlangsung sampai
membentuk J yang valid. Komponen jam dikatakan valid,
   Jika : 0<=H<=23, 0<=M<=60 dan 0<=S<=59
*/
int hh, mm, ss;
do {
scanf("%d", &hh);
scanf("%d", &mm);
scanf("%d", &ss);
}while (hh<0 || hh>23 || mm<0 || mm>59 || ss<0 || ss>59);
CreateJam(&(*J), hh, mm, ss);
}

void PrintJam (JAM J) {
/* I.S. : J diisi dengan nilai sembarang
   F.S. : Menampilkan jam (J) ke layar dengan format HH:MM:SS

```

```

*/
    printf("%d:%d:%d", J.Hour, J.Minute, J.Second);
}

/** { kelompok Validasi Type } */
boolean IsJValid (int H, int M, int S) {
/*  fungsi pengecekan nilai komponen jam yang valid
    Mengirimkan true jika H,M,S dapat membentuk J yang valid
    Sebaliknya mengirimkan false jika H,M,S tidak dapat membentuk J yang
valid
*/
    if (H>=0 && H<=23 && M>=0 && M<=59 && S>=0 && S<=59)
        return (true);
    else
        return (false);
}

/** {Operator Relasional} */
boolean JEQ(JAM J1, JAM J2) {
/* Mengirimkan true jika J1=J2, false jika tidak
Artinya mengirimkan true jika semua elemen jam bernilai bernilai sama
Contoh :
    - Jika J1 = 10:20:30 dan J2 = 10:20:30, maka mengirimkan true
    - Jika J1 = 10:20:30 dan J2 = 10:30:30, maka mengirimkan false
*/
    if (J1.Hour == J2.Hour && J1.Minute == J2.Minute && J1.Second ==
J2.Second)
        return (true);
    else
        return (false);
}

boolean JLT(JAM J1, JAM J2) {
/* Mengirimkan true jika J1<J2 , false jika tidak
Artinya mengirimkan true jika elemen pada J1 bernilai kurang dari J2
Contoh :
    - Jika J1 = 10:20:30 dan J2 = 10:30:30, maka mengirimkan true
    - Jika J1 = 10:30:30 dan J2 = 10:30:30, maka mengirimkan false
    - Jika J1 = 10:30:30 dan J2 = 11:30:30, maka mengirimkan false
*/
    if (J1.Hour < J2.Hour || J1.Minute < J2.Minute || J1.Second < J2.Second)

```

```

        return (true);
    else
        return (false);
}

boolean JGT(JAM J1, JAM J2) {
/* Mengirimkan true jika J1>J2, false jika tidak
Artinya mengirimkan true jika elemen pada J1 bernilai lebih dari J2
Contoh :
    - Jika J1 = 10:40:30 dan J2 = 10:30:30, maka mengirimkan true
    - Jika J1 = 10:30:30 dan J2 = 10:30:30, maka mengirimkan false
    - Jika J1 = 10:20:30 dan J2 = 10:20:30, maka mengirimkan false
*/
    if (J1.Hour > J2.Hour || J1.Minute > J2.Minute || J1.Second > J2.Second)
        return (true);
    else
        return (false);
}

/** { operator aritmatika } */
void Reset (JAM *J) {
/* I.S. : J sembarang
F.S. : J bernilai "nol", yaitu semua komponen bernilai 0
Proses : Ubah semua nilai elemen J menjadi 0:0:0
*/
    (*J).Hour = 0;
    (*J).Minute = 0;
    (*J).Second = 0;
}

JAM NextDetik (JAM J) {
/* Menambahkan nilai 1 detik pada jam, dan kirimkan nilai jam baru */
    JAM jamBaru;
/* mencopy nilai J kepada jamBaru {bisa pakai cara ini atau yang dibawah}
jamBaru.Hour = J.Hour;
jamBaru.Minute = J.Minute;
jamBaru.Second = J.Second + 1;
*/
    jamBaru = J;
    jamBaru.Second = jamBaru.Second + 1;
    if (jamBaru.Second > 59) {

```

```

    jamBaru.Second = 0;
    jamBaru.Minute = jamBaru.Minute + 1;
    if (jamBaru.Minute > 59) {
        jamBaru.Minute = 0;
        jamBaru.Hour = jamBaru.Hour + 1;
        if(jamBaru.Hour > 23){
            jamBaru.Hour = 0;
        }
    }
}

return (jamBaru);
}

JAM NextNDetik (JAM J, int N) {
/* Menambahkan nilai N detik pada jam, dan mengirimkan nilai jam baru */
int i;
JAM jamBaru;

jamBaru = J;
if (N > 0 && N <= 86400) { // menambahkan detik hanya sampai 24 jam
    for (i=1; i<=N; i++){
        jamBaru = NextDetik (jamBaru);
    }
}

return (jamBaru);
}

long int Durasi ( JAM JAww , JAM JAkh) {
/* Menghitung selisih dari dua buah jam dalam bentuk detik
    Proses : Mengirimkan JAkh -JAww dlm Detik, Hasilnya negatif jika JAww >
    JAakhir
    Jika hasilnya negatif maka ubahlah menjadi nilai positif
*/
long int detikAwl, detikAkh, result;

detikAwl = JamToDetik(JAww);
detikAkh = JamToDetik(JAkh);
result = detikAwl - detikAkh;
if (result < 0) {

```

```

        result = result * (-1);
    }
    return (result);
}
/** {Kelompok Konversi Terhadap Type} **/
long int JamToDetik (JAM J) {
    /* melakukan konversi jam menjadi Detik
        Rumus : detik = 3600*hour+menit*60 + detik
        nilai maksimum = 3600*23+59*60+59*60
    */
    long int detik;
    detik = (J.Hour * 3600) + (J.Minute * 60) + J.Second;
    return (detik);
}

```

I.1.D. Source Code mjam.c

```

/*
    Program      : mjam.c
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas        : C
    Deskripsi    : contoh main driver dari ADT Jam
    tanggal      : 15-03-2024
*/

#include "jam.c"
#include <stdio.h>
#include <conio.h>

int main() {
    // Kamus Data
    JAM J1, J2, J3, J4;
    int tambahanDetik;

    // Algoritma
    ReadJam(&J1);

    printf("Nilai J1 = ");
    PrintJam(J1);
    printf("\n");
}

```

```

J2 = NextDetik(J1);
printf("Nilai J2 = ");
PrintJam(J2);
printf("\n");

printf("Masukan nilai N detik (0 .. 59) : ");
scanf("%d", &tambahanDetik);

J3 = NextNDetik(J1, tambahanDetik);
printf("Nilai J3 = ");
PrintJam(J3);
printf("\n");

J4 = J1;
printf("Nilai J4 = ");
PrintJam(J4);
printf("\n");

printf("\nPengecekan Operator Relasional\n");
if (JEQ(J1, J2)) {
    printf("nilai J1 sama dengan J2\n");
}

if (JLT(J1, J2)) {
    printf("nilai J1 lebih kecil dari J2\n");
}

if (JGT(J1, J2)) {
    printf("nilai J1 lebih besar dari J2\n");
}

if (JEQ(J3, J1)) {
    printf("nilai J3 sama dengan J1\n");
}

if (JLT(J3, J1)) {
    printf("nilai J3 lebih kecil dari J1\n");
}

if (JGT(J3, J1)) {
    printf("nilai J3 lebih besar dari J1\n");
}

```

```

    }

    if (JEQ(J1, J4)) {
        printf("nilai J1 sama dengan J4\n");
    }

    if (JLT(J1, J4)) {
        printf("nilai J1 lebih kecil dari J4\n");
    }

    if (JGT(J1, J4)) {
        printf("nilai J1 lebih besar dari J4\n");
    }

    Reset(&J4);
    printf("nilai J4 = ");
    PrintJam(J4);
    printf("\n");

    return 0;
}

```

I.1.E. Hasil

```

C boolean.h M  C jam.h M  C jam.c A  C mjam.c A x  C jam.h (Working Tree) M
C mjam.c > ...
1 /*
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT jam> cd 'd:\pemrograman\aji\prakStrukDat\pertemuan1\ADT jam\output'
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT jam\output> & .\mjam.exe
18
8
0
Nilai J1 = 18:8:0
Nilai J2 = 18:8:1
Masukan nilai N detik (0 .. 59) : 15
Nilai J3 = 18:8:15
Nilai J4 = 18:8:0

Pengecekan Operator Relasional
nilai J1 lebih kecil dari J2
nilai J3 lebih besar dari J1
nilai J1 sama dengan J4
nilai J4 = 0:0:0
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT jam\output>

```

Gambar I.1 Output Program ADT Jam

I.1.F. Analisa

Program ini adalah program untuk merepresentasikan waktu dalam format jam, menit, detik. ADT jam ini biasanya menyediakan operasi-operasi untuk mengatur waktu, serta operasi-operasi untuk mendapatkan nilai jam, menit, detik. Dengan menggunakan program ADT Jam ini, kita dapat dengan mudah mengelola dan memanipulasi waktu dalam program tanpa mereka perlu memikirkan detail implementasi internal dari representasi waktu.

Ada beberapa program yang di buat akan di jelaskan seperti Boolean.h, jam.h, jam.c, mjam.c yang dibuat untuk menciptakan satu program utama yaitu ADT Jam:

- i. Boolean.h disini menjelaskan beberapa definisi untuk membuat header Boolean. Dalam kode tersebut, boolean didefinisikan sebagai tipe data unsigned char, dan nilai true diatur sebagai 1 sedangkan nilai false diatur sebagai 0. Ini adalah cara yang umum digunakan untuk mendefinisikan tipe data boolean dalam bahasa pemrograman C di lingkungan yang tidak mendukung tipe data boolean bawaan. Dengan definisi ini, Anda dapat menggunakan true dan false seperti tipe data boolean standar dalam bahasa pemrograman C.
- ii. Jam.h adalah header dari prototype jam, definisi struktur data JAM dan beberapa fungsi yang berkaitan dengan ADT Jam. ADT Jam digunakan untuk merepresentasikan waktu dalam format jam, menit, dan detik. Beberapa fungsi yang disediakan antara lain konstruktor, selektor, operator relasional, operator aritmatika, dan konversi terhadap tipe data. Fungsi-fungsi tersebut dapat digunakan untuk membaca, menulis, dan memanipulasi waktu dalam format JAM.
- iii. Jam.c adalah program yang berisi implementasi prosedur dan fungsi yang sudah didefinisikan pada program jam.h. operasi ini melakukan beberapa perintah yaitu inisialisasi jam, mengurangi atau menambahkan waktu jam, dan lain lain
- iv. Mjam.c ini adalah sebuah program utama yang dibuat dengan definisi-definisi, prosedur maupun fungsi yang telah dikerjakan dari file program sebelumnya. Program ini akan menjalankan ADT Jam untuk membuat jam baru, menambahkan waktu, menampilkannya, dan lain lain. Program mjam.c ini akan mengeluarkan output program sesuai yang telah kita isikan pada waktu (jam, menit, detik) berapa sesuai dengan prosedur dan fungsi yang telah dipanggil dari file jam.c.

BAB II. TUGAS PRAKTIKUM

II.1 Tugas ADT Point

II.1.A. Source Code Boolean.h

```
#ifndef boolean_H
#define boolean_H
#define true 1
#define false 0
#define boolean unsigned char

#endif
```

II.1.B. Source Code Point.h

```
/*
    Program      : point.h
    Author       : 2350081062, Aji Kartiko Hartanto
    Kelas        : C
    Deskripsi    : Header file dari prototype point
    Tanggal      : 16-03-2024
*/

#ifndef _POINT_H
#define _POINT_H

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

#define Absis(P) (P).X
#define Ordinat(P) (P).Y
#define PI 3.14159265

// Definisi ABSTRACT DATA TYPE POINT
typedef struct {
    int X;
    int Y;
} POINT;
```

```

/* Prototype POINT */

/* Konstruktor membentuk POINT */
void CreatePOINT (POINT *P);
/* I.S : P terdefinisi, tidak diketahui nilainya
   F.S : membentuk sebuah POINT dari komponen-komponennya dengan P
   diinisialisasi nilainya dengan X=0 dan Y=0
*/

void CreatePOINT2 (POINT *P, int XBaru, int YBaru);
/* I.S : P terdefinisi, tidak diketahui nilainya
   F.S : membentuk sebuah POINT dari komponen-komponennya dengan P
   diinisialisasi nilainya dengan X= XBaru dan Y=YBaru
*/

/* Selektor POINT **/
int GetAbsis(POINT P);
// Mengirimkan komponen Absis dari P
int GetOrdinat (POINT P);
// Mengirimkan komponen Ordinat dari P POINT

/* Set nilai komponen */
void SetAbsis(POINT *P, int newX);
/* I.S : P terdefinisi, dengan nilainya sembarang
   F.S : Mengubah nilai komponen Absis dari P dengan X=newX
*/
void SetOrdinat (POINT *P, int newY);
/* I.S : P terdefinisi, dengan nilainya sembarang
   F.S : Mengubah nilai komponen Ordinat dari P dengan Y=newY
*/

/* Destruktor/Dealokator: tidak perlu */

/** { KELOMPOK Interaksi dengan I/O device, BACA/TULIS } **/
void BacaPOINT (POINT *P);
/* I.S : P terdefinisi, mungkin kosong
   F.S : P terdefinisi, dengan membaca nilai X dan Y
*/
void CetakPOINT(POINT P);
/* I.S : P terdefinisi, mungkin kosong
   F.S : Menampilkan nilai komponen P ke layar dg format "(X , Y)
*/

```

```

/** Kelompok operasi relasional terhadap POINT */
boolean EQ(POINT P1, POINT P2);
// Mengirimkan true jika P1 = P2, dan false jika sebaliknya
boolean NEQ(POINT P1, POINT P2);
// Mengirimkan true jika P1 tidak sama dengan P2, dan false jika sebaliknya
boolean LT(POINT P1, POINT P2);
// Mengirimkan true jika P1 < P2, dan false jika sebaliknya
// Definisi lebih kecil: posisi titik lebih ke kiri atau ke bawah dalam
bidang kartesian
boolean MT(POINT P1, POINT P2);
// Mengirimkan true jika P1 > P2, dan false jika sebaliknya.
// Definisi lebih besar: posisi titik lebih ke kanan atau ke atas dalam
bidang kartesian

/** Kelompok menentukan di mana P berada */
boolean IsOrigin (POINT P);
// Menghasilkan true jika P berada pada titik origin yaitu nilai X=0 dan
Y=0, dan false jika sebaliknya
boolean IsOnSbX (POINT P);
// Menghasilkan true jika P terletak Pada sumbu X yaitu nilai Y=0, dan
false jika sebaliknya
boolean IsOnSbY (POINT P);
// Menghasilkan true jika P terletak pada sumbu Y yaitu nilai X=0, dan
false jika sebaliknya
int Kuadran(POINT P);
// Menghasilkan kuadran dari P: 1,2,3, atau 4
// Precondition : P bukan Titik Origin, dan P tidak terletak di salah satu
sumbu

/** KELOMPOK OPERASI LAIN TERHADAP TYPE */
POINT MirrorOf(POINT P, boolean SbX, boolean SbY);
// Menghasilkan salinan P yang dicerminkan tergantung nilai SbX dan SBY
// Jika SbX bernilai true, maka dicerminkan terhadap Sumbu X
// Jika SbY bernilai true, maka dicerminkan terhadap Sumbu Y
double JarakO(POINT P);
// Menghitung jarak P dari titik origin (0,0)
void GeserKeSbX (POINT *P);
/* I.S : P terdefinisi, mungkin kosong
   F.S : P berada pada Sumbu X, jika tidak berada pada Sumbu X maka geser P
ke Sumbu X.
Contoh: Jika koordinat semula(9,9) menjadi (9,0)

```

```

*/
void GeserKeSbY(POINT *P);
/* I.S : P terdefinisi, mungkin kosong
   F.S : P berada pada Sumbu Y, jika tidak berada pada Sumbu Y maka geser P
   ke Sumbu Y.
   Contoh: Jika koordinat semula(9,9) menjadi (0,9)
*/

#endif

```

II.1.C. Source Code Point.c

```

/*
   Program      : point.c
   Author       : 2350081062, Aji Kartiko Hartanto
   Kelas        : C
   Deskripsi    : Body file dari prototype point
   Tanggal     : 16-03-2024
*/

#include "point.h"
#include <stdio.h>
#include <conio.h>
#include <math.h>

/* Prototype POINT */
void CreatePoint(POINT *P) {
    (*P).X = 0;
    (*P).Y = 0;
}
/*
   I.S : P terdefinisi, tidak diketahui nilainya
   F.S : membentuk sebuah POINT dari komponen-komponennya dengan P
   diinisialisasi nilainya dengan X=0 dan Y=0
*/

void CreatePoint2(POINT *P, int XBaru, int YBaru) {
    (*P).X = XBaru;
    (*P).Y = YBaru;
}
/*

```

```

        I.S : P terdefinisi, tidak diketahui nilainya
        F.S : membentuk sebuah POINT dari komponen-komponennya dengan P
diinisialisasi nilainya dengan X=XBaru dan Y=YBaru
*/

/* Selektor POINT */
int GetAbsis(POINT P) {
    return P.X;
}
//Mengirimkan komponen Absis dari P

int GetOrdinat(POINT P) {
    return P.Y;
}
//Mengirimkan komponen Ordinat dari P POINT

/* Set nilai komponen */
void SetAbsis(POINT *P, int newX) {
    (*P).X = newX;
}
/*
        I.S : P terdefinisi, dengan nilainya sembarang
        F.S : Mengubah nilai komponen Absis dari P dengan X=newX
*/

void SetOrdinat(POINT *P, int newY) {
    (*P).Y = newY;
}
/*
        I.S : P terdefinisi, dengan nilainya sembarang
        F.S : Mengubah nilai komponen Absis dari P dengan Y=newY
*/

/* Destruktor/Dealokator: tidak perlu */

/* {Kelompok Interaksi dengan I/O device, BACA/TULIS} */
/*
        I.S : P terdefinisi, kemungkinan kosong
        F.S : P terdefinisi, dengan membaca nilai X dan Y
*/
void BacaPoint(POINT *P) {

```

```

// kamus lokal
int x, y;

// algoritma
printf("Masukan Absis: ");
scanf("%d", &x);

printf("Masukan Ordinat: ");
scanf("%d", &y);

CreatePoint2(&(*P), x, y);
}

/*
    I.S : P terdefinisi, mungkin kosong
    F.S : Menampilkan nilai komponen P ke layar dengan format "(X,Y)"
*/
void CetakPoint(POINT P) {
    printf("titik : (%d, %d)", GetAbsis(P), GetOrdinat(P));
}

/* Kelompok operasi relasional terhadap Point */
boolean EQ(POINT P1, POINT P2) {
    if ((GetAbsis(P1) == GetAbsis(P2)) && (GetOrdinat(P1) ==
GetOrdinat(P2))) {
        return true;
    } else {
        return false;
    }
}

//Mengirimkan true jika P1 = P2, dan false jika sebaliknya

boolean NEQ(POINT P1, POINT P2) {
    if ((GetAbsis(P1) != GetAbsis(P2)) || (GetOrdinat(P1) !=
GetOrdinat(P2))) {
        return true;
    } else {
        return false;
    }
}

//Mengirimkan true jika P1 tidak sama dengan P2, dan false jika sebaliknya

```

```

boolean LT(POINT P1, POINT P2) {
    if ((GetAbsis(P1) < GetAbsis(P2)) || (GetOrdinat(P1) < GetOrdinat(P2)))
    {
        return true;
    } else {
        return false;
    }
}
//Mengirimkan true jika P1 < P2, dan false jika sebaliknya definisi lebih
kecil: posisi titik lebih ke kiri atau ke bawah dalam bidang kartesian

boolean MT(POINT P1, POINT P2) {
    if ((GetAbsis(P1) > GetAbsis(P2)) || (GetOrdinat(P1) > GetOrdinat(P2)))
    {
        return true;
    } else {
        return false;
    }
}
//Mengirimkan true jika P1 > P2, dan false jika sebaliknya definisi lebih
besar: posisi titik lebih ke kanan atau ke atas dalam bidang kartesian

/* Kelompok menentukan di mana P berada */
boolean IsOrigin(POINT P) {
    if ((GetAbsis(P) == 0) && (GetOrdinat(P) == 0)) {
        return true;
    } else {
        return false;
    }
}
//Menghasilkan true jika P berada pada titik origin yaitu nilai X=0 dan
Y=0, dan false jika sebaliknya

boolean IsOnSbX(POINT P) {
    if ((GetOrdinat(P) == 0)) {
        return true;
    } else {
        return false;
    }
}

```

```

//Menghasilkan true jika P terletak Pada sumbu X yaitu nilai Y=0, dan false
jika sebaliknya

boolean IsOnSbY(POINT P) {
    if ((GetAbsis(P) == 0)) {
        return true;
    } else {
        return false;
    }
}

//Menghasilkan true jika P terletak Pada sumbu Y yaitu nilai X=0, dan false
jika sebaliknya

int Kuadran(POINT P) {
    if ((GetAbsis(P) > 0) && (GetOrdinat(P) > 0)) {
        return 1;
    } else if ((GetAbsis(P) < 0) && (GetOrdinat(P) > 0)) {
        return 2;
    } else if ((GetAbsis(P) < 0) && (GetOrdinat(P) < 0)) {
        return 3;
    } else if ((GetAbsis(P) > 0) && (GetOrdinat(P) < 0)) {
        return 4;
    } else {
        return 0;
    }
}

/* Menghasilkan kuadran dari P: 1,2,3, atau 4
   Precondition : P bukan Titik Origin, dan P tidak terletak di salah satu
sumbu
*/

/* Kelompok operasi lain terhadap type */
POINT MirrorOf(POINT P, boolean SbX, boolean SbY) {
    // kamus lokal
    POINT newPoint;

    // algoritma
    CreatePoint2(&newPoint, GetAbsis(P), GetOrdinat(P));

    if (SbX == true) {
        newPoint.X = GetAbsis(P) * -1;
    }
}

```



```

    } else if (SbY == true) {
        newPoint.Y = GetOrdinat(P) * -1;
    }

    return newPoint;
}
/* Menghasilkan salinan P yang dicerminkan tergantung nilai SbX dan SBY
    -Jika SbX bernilai true, maka dicerminkan terhadap Sumbu X
    -Jika SbY bernilai true, maka dicerminkan terhadap Sumbu Y
*/

//Menghitung jarak P dari titik Origin (0, 0)
double JarakO(POINT P) {
    // kamus lokal
    double jarak;

    // algoritma
    /*
    gunakan function sqrt() yang diambil dari library math.h
    function sqrt digunakan sebagai akar dari matematika dan memiliki 1
parameter

    gunakan function pow() yang diambil dari library math.h
    function pow digunakan sebaga exponensial dari matematika dan memiliki
2
parameter, parameter yang pertama digunakan sebagai base, dan parameter
kedua digunakan sebagai exponen
    */
    jarak = sqrt(pow((GetAbsis(P) - 0), 2) + pow((GetOrdinat(P) - 0), 2));

    return jarak;
}

void GeserKeSbX(POINT *P) {
    if (GetOrdinat((*P)) != 0) {
        SetOrdinat(&(*P), 0);
    }
}
/*
I.S : P terdefinisi, mungkin kosong

```

F.S : P berada pada Sumbu X, jika tidak berada pada Sumbu X maka geser P ke Sumbu X.

Contoh: Jika koordinat semula(9,9) menjadi (9,0)

*/

```
void GeserKeSbY(POINT *P) {  
    if (GetAbsis((*P)) != 0) {  
        SetAbsis(&(*P), 0);  
    }  
}
```

/*

I.S : P terdefinisi, mungkin kosong

F.S : P berada pada Sumbu Y, jika tidak berada pada Sumbu Y maka geser P ke Sumbu Y.

Contoh: Jika koordinat semula(9,9) menjadi (0,9)

*/

II.1.D. Source Code mpoint.c

/*

```
Program      : mpoint.c  
Author       : 2350081062, Aji Kartiko Hartanto  
Kelas       : C  
Deskripsi    : Main driver dari ADT Point  
Tanggal      : 16-03-2024
```

*/

```
#include "point.c"  
#include <stdio.h>  
#include <conio.h>
```

```
int main() {  
    // Kamus  
    POINT P1, P2, newPoint;  
  
    // Algoritma  
    // membuat point  
    CreatePoint(&P1);  
    CreatePoint(&newPoint);  
    CreatePoint2(&P2, 5, 7);  
  
    // membaca point 1
```

```

BacaPoint(&P1);

// cetak point 1 dan point 2
CetakPoint(P1);
printf("\n");
CetakPoint(P2);

// Mengambil nilai absis dan ordinat dari kedua point (P1, P2)
printf("\n");
printf("\nP1 Absis : %d", GetAbsis(P1));
printf("\nP1 Ordinat : %d", GetAbsis(P1));
printf("\nP2 Absis : %d", GetAbsis(P2));
printf("\nP2 Ordinat : %d", GetAbsis(P2));
printf("\n");

// Megubah nilai absis dan ordinat dari kedua point (P1, P2)
SetAbsis(&P1, -4);
SetOrdinat(&P1, -7);
SetAbsis(&P2, 7);
SetOrdinat(&P2, -4);

printf("\n");
CetakPoint(P1);
printf("\n");
CetakPoint(P2);
printf("\n");

// Operasi relasional
if (EQ(P1, P2) == true) {
    printf("\ntrue");
} else {
    printf("\nfalse");
}

if (NEQ(P1, P2) == true) {
    printf("\ntrue");
} else {
    printf("\nfalse");
}

if (LT(P1, P2) == true) {

```

```

        printf("\ntrue");
    } else {
        printf("\nfalse");
    }

    if (MT(P1, P2) == true) {
        printf("\ntrue");
    } else {
        printf("\nfalse");
    }

    // Menentukan P berada pada kuadran apa
    printf("\n");
    if (IsOrigin(P1) == true) {
        printf("\nPoint P1 tidak berada pada kuadran manapun\n");
    } else {
        printf("\nPoint P1 terletak di kuadran %d\n", Kuadran(P1));
    }

    if (IsOnSbX(P1) == true) {
        printf("\nP1 berada pada sumbu X\n");
    } else if (IsOnSbY(P1) == true) {
        printf("\nP1 berada pada sumbu Y\n");
    } else {
        printf("\nP1 tidak terletak pada sumbu apapun\n");
    }

    // Operasi lain terhadap type
    printf("\nJarak P1 dengan origin: %.2f\n", JarakO(P1));

    GeserKeSbX(&P1);
    CetakPoint(P1);
    printf("\n");

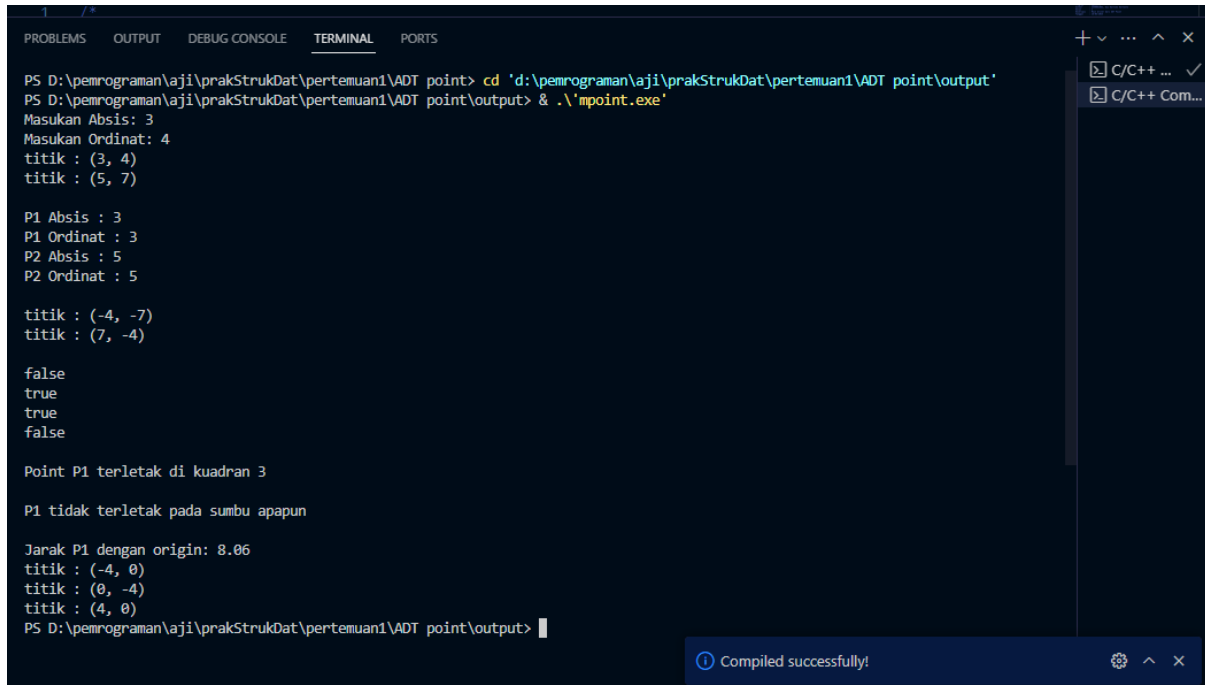
    GeserKeSbY(&P2);
    CetakPoint(P2);
    printf("\n");

    newPoint = MirrorOf(P1, IsOnSbX(P1), IsOnSbY(P1));
    CetakPoint(newPoint);
    printf("\n");

```

```
    return 0;
}
```

II.1.E. Hasil



```
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT point> cd 'd:\pemrograman\aji\prakStrukDat\pertemuan1\ADT point\output'
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT point\output> & .\mpoint.exe
Masukan Absis: 3
Masukan Ordinat: 4
titik : (3, 4)
titik : (5, 7)

P1 Absis : 3
P1 Ordinat : 3
P2 Absis : 5
P2 Ordinat : 5

titik : (-4, -7)
titik : (7, -4)

false
true
true
false

Point P1 terletak di kuadran 3

P1 tidak terletak pada sumbu apapun

Jarak P1 dengan origin: 8.06
titik : (-4, 0)
titik : (0, -4)
titik : (4, 0)
PS D:\pemrograman\aji\prakStrukDat\pertemuan1\ADT point\output>
```

Gambar II.1 Output Program Tugas ADT Point

II.1.F. Analisa

ADT Point ini adalah sebuah tipe data yang merepresentasikan sebuah titik dalam bidang kartesian dengan dua komponen, yaitu absis (X) dan ordinat (Y). Program ADT Point ini juga mirip seperti program pada Latihan di bab 1, dibuat dengan beberapa file untuk melakukan operasi point. Yang pertama adalah file Boolean.h. didalam file program Boolean ini terdapat beberapa definisi untuk menjalankan program, dan definisi nya sama dengan ADT jam.

Selanjutnya adalah file program point.h, file program ini adalah header dari prototype point, didalam program ini beberapa fungsi di deklarasikan untuk membuat program dari ADT Point, contoh nya adalah

- i. Fungsi CreatePOINT dan CreatePOINT2 digunakan untuk membuat sebuah objek Point dengan menginisialisasi nilai Absis dan Ordinat dari parameter yang diberikan. Fungsi ini memastikan bahwa nilai Absis dan Ordinat yang diberikan valid untuk membentuk sebuah Point.

- ii. Fungsi GetAbsis dan GetOrdinat digunakan untuk mengambil nilai Absis dan Ordinat dari sebuah objek Point.
- iii. Fungsi SetAbsis dan SetOrdinat digunakan untuk mengubah nilai Absis dan Ordinat dari sebuah objek Point dengan nilai baru yang diberikan sebagai parameter.
- iv. Fungsi BacaPOINT dan CetakPOINT digunakan untuk membaca dan menampilkan nilai Absis dan Ordinat dari sebuah objek Point.
- v. Fungsi-fungsi EQ, NEQ, LT, dan MT digunakan untuk membandingkan dua objek Point berdasarkan posisi relatifnya dalam bidang kartesian.

Selanjutnya file program point.c digunakan untuk implementasi program ADT Jam, dan cara kerjanya juga hampir mirip dengan program ADT jam pada Latihan jam.c tetapi disini point.c memiliki cara kerja yang berbeda, tidak seperti jam.c.

Mpoint.c adalah file main driver, atau program utama untuk membuat ADT Point. Tipe data ini memiliki dua atribut, yaitu koordinat x dan y. setelah kita melakukan deklarasi tipe data point, selanjutnya adalah kita menggunakan fungsi fungsi seperti CreatePoint(), fungsi main, dan lain lain.

Jadi dalam program diatas, kita dapat membuat point baru dengan menggunakan beberapa fungsi yang telah di jelaskan , kita juga dapat mengubah koordinat point dengan cara mengakses atribut x dan y pada variable point dan menampilkannya Kembali.

BAB III. KESIMPULAN

Dalam praktikum ini, mahasiswa dapat memahami bagaimana sebuah tipe data abstrak dapat direpresentasikan dan dioperasikan dalam bahasa pemrograman, serta bagaimana struktur data dapat digunakan untuk merepresentasikan entitas matematis seperti waktu (jam) dan titik dalam koordinat dua dimensi. Hal ini memberikan dasar yang kuat bagi mahasiswa dalam memahami dan mengembangkan aplikasi yang memanfaatkan konsep-konsep tersebut.

Dengan demikian, praktikum struktur data mengenai ADT jam dan ADT point memberikan pemahaman yang baik tentang konsep ADT, penggunaan struktur data, penerapan konsep OOP, dan penerapan ADT dalam pengembangan program.