Modul Praktikum
Algoritma & Struktur Data

Modul pembelajaran untuk praktikum Algoritma dan Struktur Data – Gasal, 2024/2025

Disusun oleh: Ozzi Suria, S.T., M.T.

Silabus Praktikum

Prak.	Materi	Waktu
1	Tipe Data, Variabel, dan Operator	120 menit
2	Kondisi dan Perulangan	120 menit
3	Fungsi dan Prosedur	120 menit
4	Array dan Matriks	120 menit
5	Record	120 menit
6	Abstract Data Type (ADT)	120 menit
7	Array with ADT	120 menit
8	Stack	120 menit
9	Queue	120 menit
10	Pointer	120 menit
11	Linked List Linear	120 menit
12	Linked List Circular	120 menit

Daftar Isi

PERTEMUAN I | TIPE DATA, VARIABEL, DAN OPERATOR

1.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 1, mahasiswa diharapkan dapat:

a. Memahami tipe data dasar dalam C.
b. Memahami penggunaan tipe data dan variabel.
c. Membuat program yang mengimplementasikan operasi antar variabel dengan tipe data yang berbeda.

1.2 Materi Praktikum

1.2.1 Tipe Data

Ada tipe data dasar yang dapat digunakan dalam bahasa C, yaitu:

1. Char : digunakan untuk menuliskan karakter
2. Integer : digunakan untuk bilangan bulat
3. Float dan Double ; digunakan untuk bilangan real

Perhatikan tabel 1. Tabel 1 berisi format tipe data yang digunakan untuk membaca/menulis nilai dari/ke variabel untuk setiap tipe data.

Tabel 1. Format Tipe Data

Tipe Data	Format
int, bool	%d
char	%c
string (array of char)	%s
Float	%f
Double	%lf

1.2.2 Variabel

Dalam Bahasa C bersifat case-sensitive (penulisan nama dengan huruf besar dan huruf kecil dibedakan). Contoh:

- Nama Temp berbeda dengan temp.

- Var_a berbeda dengan var_a, Var_A

Konsistensi penulisan nama (nama fungsi, nama prosedur, nama variabel) wajib diperhatikan! Pemanggilan nama variabel yang salah akan memberikan hasil error pada program.

1.2.3 Operator

Operasi yang dapat dilakukan oleh setiap tipe data berbeda-beda. Untuk lebih jelasnya perhatikan tabel 2.

Tabel 2. Operasi Tipe Data

Tipe Data	Operasi
bool	Logika (not, and, or, xor)
int	Aritmatika (+, -, *, /, %) dan Perbandingan (=, !=, >, <, <=, >=)
char	Concatenation (+)
string (array of char)	Concatenation (+)
float	Aritmatika (+, -, *, /) dan Perbandingan (!=, >, <, <=, >=)
double	Aritmatika (+, -, *, /) dan Perbandingan (!=, >, <, <=, >=)

1.3 Guided

Kerjakan dan pelajari kode yang ada pada setiap latihan berikut!

Guided 1

```
#include<stdio.h>
int main()
{
	int umur;
	printf("Hello World!\n");
	printf("Input umur anda: ");scanf("%d",&umur);
	printf("Umur anda: %d",umur);
	getch();
	return 0;
}
```

Guided 2

```
#include <stdio.h>

int main()
{
	float berat, gram, ton, pound;
	printf("Input Berat Anda dalam kilogram: ");scanf("%f",&berat);
	gram=berat*1000;
	ton=berat/1000;
	pound=berat*2.20462;
	printf("Berat anda dalam gram adalah %f\n",gram);
	printf("Berat anda dalam ton adalah %f\n",ton);
	printf("Berat anda dalam pound adalah %f",pound);
	getch();
	return 0;
}
```

PERTEMUAN II | KONDISI DAN PERULANGAN

1.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 2, mahasiswa diharapkan dapat:

a. Memahami penggunaan kondisi dan perulangan dalam algoritma pemrograman.
b. Menganalisa kebutuhan kondisi dan perulangan yang diperlukan dari kasus yang ada
c. Mengimplementasikan kondisi dan perulangan dalam algoritma

1.2 Materi Praktikum

1.2.1 Kondisi

Dalam pernyataan kondisional, sebuah aksi dikerjakan apabila kondisi dalam instruksi tersebut terpenuhi. Pengerjaan kondisi *if* dalam algoritma sesuai dengan urutannya. Contoh pernyataan kondisional adalah:

```
if(x%2==0){
    printf("Bilangan Genap!");
}else{
    printf("Bilangan Ganjil!");
}
```

Operasi perbandingan untuk kondisi pernyataan kondisional dapat menggunakan syntax berikut:

Operasi	Symbol	Syntax in C
Equal (sama dengan)	=	==
Not Equal (tidak sama dengan)	≠	!=
And (dan)	&	&&
Or (atau)		||

Untuk operasi perbandingan yang lain menggunakan tanda sesuai dengan ketentuannya masing-masing.

- >= → lebih besar sama dengan
- <= → lebih kecil sama dengan
- < → lebih kecil
- > → lebih besar

1.2.2 Perulangan

Perulangan dalam suatu algoritma memungkinkan komputer untuk dapat mengerjakan instruksi secara berulang. Ada 3 perulangan yang dapat digunakan yaitu *for, while,* dan *do-while*.

Contoh perulangan for:

```
int i;
for(i=0;i<10;i++){
    printf("Perulangan ke %d, angka %d",i,i+1);
}
```

Contoh perulangan while:

```
int i=15;
while(i>=0)
{
    if(i%2==1){
      printf("Bilangan Ganjil");
    }else{
      printf("Bilangan Genap");
    }
    i--;
}
```

Contoh perulangan do-while:

```
int x;
do{
    printf("Masukkan bilangan x: ");scanf("%d",&x);
}while(x%2==0);
printf("Program berhenti karena %d adalah bilangan ganjil",x);
```

1.3 Guided

Kerjakan dan pelajari kode yang ada pada setiap latihan berikut!

Guided 1

```
#include <stdio.h>

void main()
{
      int i, counter=0, bil;
      printf("Input Bilangan: ");scanf("%d",&bil);
      for(i=0;i<=bil;i++){
            if(i%2==0){
```

```
                counter++;
            }
        }
        printf("Dari 0 sampai %d terdapat %d bilangan genap\n",bil,counter);
}
```

Guided 2

```
#include <stdio.h>

void main()
{
        int i, counter=0, bil;
        printf("Input Bilangan: ");scanf("%d",&bil);
        for(i=0;i<=bil;i++){
            if(i!=0){
                if(bil%i==0){
                    counter++;

                }
            }
        }
        if(counter==2){
            printf("Bilangan %d adalah bilangan prima\n",bil);
        }else{
            printf("Bilangan %d bukan bilangan prima\n",bil);
        }
}
```

Guided 3

```
#include <stdio.h>

void main()
{
        int i, j, counter=0, bil;
        printf("Input Bilangan: ");scanf("%d",&bil);
        printf("Bilangan Prima 0-%d: \n",bil);
        for(i=0;i<=bil;i++){
            counter=0;
            if(i!=0){
                j=0;
                while(j<=i){
```

```
                        if(j!=0){
                              if(i%j==0){
                                    counter++;

                              }
                        }
                        j++;
                  }
            }
            if(counter==2){
                  printf("%d ",i);
            }
      }
}
```

PERTEMUAN III | FUNGSI DAN PROSEDUR

3.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 3, mahasiswa diharapkan dapat:

a. Memahami implementasi prosedur dan fungsi dalam pemrograman
b. Memahami proses pertukaran data yang terjadi melalui parameter pada fungsi dan prosedur.
c. Menganalisa masalah dan menentukan implementasi prosedur atau fungsi yang tepat untuk pemrograman.

3.2 Materi Praktikum

3.2.1 Prosedur

Prosedur adalah submodul dalam algoritma yang berisi sekumpulan instruksi yang akan dijalankan. Prosedur tidak memiliki nilai balikan sehingga tipe balikannya adalah *void*. Setelah semua instruksi di dalam prosedur selesai diproses maka dilanjutkan kembali memabaca instruksi selanjutnya di badan program pemanggil prosedur.

Contoh prosedur:

```
void hitungLuasBalok(float p, float l, float, t)
{
    float volume;
    volume=p*l*t;
    printf("Volume Balok: %d",volume);
}
```

3.2.2 Fungsi

Fungsi adalah sebuah submodul dalam algoritma yang berisi sekumpulan instruksi yang akan dijalankan. Fungsi memiliki nilai balikan *(return value)* yang dapat digunakan untuk mengembalikan nilai ke badan program pemanggil fungsi tersebut setelah semua instruksi di dalam fungsi diproses. Tipe nilai balikan harus sesuai dengan tipe balikan yang dideklarasikan untuk fungsi tersebut.

Contoh fungsi:

```
float hitungLuasBalok(float p, float l, float, t)
{
    float volume;
```

```
    volume=p*l*t;
    return volume;
}
```

3.2.3 Parameter

Ada 2 macam parameter yang dapat digunakan dalam fungsi dan prosedur pada bahasa C, yaitu:

a. Parameter Input

   Parameter input adalah parameter yang hanya dapat digunakan untuk menerima nilai variable dari badan program utama ke prosedur/fungsi. Parameter input tidak dapat digunakan untuk menyimpan nilai modifikasi dari prosedur/fungsi.

b. Parameter Input/Output

   Parameter input/output adalah parameter yang dapat digunakan untuk menerima nilai variable dari badan program utama ke prosedur/fungsi. Parameter input/output dapat digunakan untuk menyimpan nilai hasil modifikasi dari prosedur/fungsi sehingga nilai tersebut dapat digunakan lagi di badan program yang memanggil fungsi.

   Penulisan parameter input/output ditandai dengan tanda asterisk (*).

Contoh penggunaan parameter input dan parameter input/output:

```
#include <stdio.h>

void hitungVolume(float p, float l, float t, float *vol);

void main()
{
    float pjg=10, lebar=2, tinggi=3, volume;
    hitungVolume(pjg, lebar, tinggi, &volume);
    printf("Volume Balok adalah: %.2f",volume);
}

void hitungVolume(float p, float l, float t, float *vol)
{
    (*vol)=p*l*t;
}
```

3.3 Guided

Kerjakan dan pelajari kode yang ada pada setiap latihan berikut!

Guided 1

```
#include <stdio.h>

void hitungKeliling(int s);
int hitungLuas(int ss_psg);
int hitungLuasPsg(int ss_psg, int *l);
void main()
{
      int pilihan, sisi, luas, luas_psg;
      do{
            system("cls");
            puts("Menu Hitung Persegi");
            puts("1. Input Sisi");
            puts("2. Lihat Sisi");
            puts("3. Hitung Keliling");
            puts("4. Hitung Luas");
            puts("5. Hitung Luas (Parameter Input/Output)");
            puts("0. Exit");
            printf("Pilih menu: ");scanf("%d",&pilihan);
            switch(pilihan)
            {
            case 1:     printf("Sisi Persegi: ");scanf("%d",&sisi);
                        getch();
                        break;
            case 2:     printf("Sisi Persegi: %d",sisi);
                        getch();
                        break;
            case 3:     hitungKeliling(sisi);
                        getch();
                        break;
            case 4:     luas=hitungLuas(sisi);
                        printf("Luas Persegi: %d",luas);
                        getch();
                        break;
            case 5:     hitungLuasPsg(sisi, &luas_psg);
                        printf("Luas Persegi: %d",luas_psg);
                        getch();
                        break;
            case 0:     break;
```

```
            default:    printf("Maaf menu tidak ditemukan!");
                        getch();
                        break;
      }while(pilihan!=0);
}
void hitungKeliling(int s)
{
      printf("Keliling Persegi: %d",4*s);
}
int hitungLuas(int ss_psg)
{
      return ss_psg*ss_psg;
}
int hitungLuasPsg(int ss_psg, int *l)
{
      (*l)=ss_psg*ss_psg;
}
```

PERTEMUAN IV | ARRAY & MATRIKS

4.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 4, mahasiswa diharapkan dapat:

a. Memahami penggunaan array dalam program.
b. Mengimplementasikan array 1 dimensi
c. Mengimplementasikan array 2 dimensi dalam bentuk matriks

4.2 Materi Praktikum

4.2.1 Array

Array atau larik adalah struktur data yang menyimpan sekumpulan elemen yang bertipe sama. Setiap elemen di dalam array dapat diakes langsung melalui indeksnya. Indeks dalam array dimulai dari 0. Jika kita mendefinisikan lebar array sebanyak 10 maka indeks yang dapat diakses untuk array tersebut adalah 0-9. Perhatikan Ilustrasi array berikut:

```
int a[5];
```

Kode tersebut berfungsi untuk mendeklarasikan variabel a sebagai array dengan lebar data 5 bertipe integer. Maka indeks array yang dapat diakses untuk variabel a adalah 0-4.

Variabel a

0	34
1	2
2	36
3	
4	

Misal:

Indeks 0 akan diisi dengan angka 34 maka pada a[0]=34;

Indeks 1 akan diisi dengan angka 2 maka a[1]=2;

Indeks 2 akan diisi dengan penjumlahan indeks 0 dan 1 maka a[2]=a[0]+a[1];

Array bersifat statis. Jumlah data array harus diketahui sebelum program dieksekusi dan jumlah elemen array tidak dapat diubah selama program dieksekusi.

Contoh akses elemen array menggunakan perulangan:

```
int i, bilangan[10];
for(i=0i<10;i++){
    bilangan[i]=i+1;
}
```

```
for(i=0i<10;i++){
    printf("Isi indeks ke %d adalah %d",i,bilangan[i]);
}
```

4.2.2 Matriks

Matriks adalah sekumpulan bilangan yang disusun dengan baris dan kolom. Dalam pemrograman matriks dapat dibentuk dengan menggunakan **array 2 dimensi**. Setiap dimensi dalam array mewakili jumlah baris dan kolom matriks secara berturut-turut. Untuk memasukkan dan mengakses setiap elemen di dalam matriks, digunakan dua perulangan. Perhatikan ilustrasi matriks 3x3 berikut:

$$\begin{matrix}(0,0) & (0,1) & (0,2)\\ (1,0) & (1,1) & (1,2)\\ (2,0) & (2,1) & (2,2)\end{matrix}$$

Setiap elemen matriks 3x3 memiliki informasi baris dan kolom. Misal setiap elemen berisi angka berikut:

$$\begin{matrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{matrix}$$

Maka dapat diketahui:

- Pada baris 0 kolom 0 terdapat elemen 1
- Pada baris 0 kolom 1 terdapat elemen 2
- Pada baris 0 kolom 2 terdapat elemen 3
- Pada baris 1 kolom 0 terdapat elemen 4
- Pada baris 1 kolom 1 terdapat elemen 5
- Pada baris 1 kolom 2 terdapat elemen 6
- Pada baris 2 kolom 0 terdapat elemen 7
- Pada baris 2 kolom 1 terdapat elemen 8
- Pada baris 2 kolom 2 terdapat elemen 9

Contoh akses matriks menggunakan perulangan:

```
int i,j, m[3][3];
for(i=0;i<3;i++){
      for(i=0;i<3;i++){
            printf("Masukkan elemen baris %d kolom %d ",i,j);scanf("%d",&m[i][j]);
      }
      printf("\n")
}
```

```
for(i=0;i<3;i++){
	for(i=0;i<3;i++){
		printf("%d ",m[i][j]);
	}
	printf("\n")
}
```

4.3 Guided

Kerjakan dan pelajari kode yang ada pada setiap latihan berikut!

Guided 1

```
#include <stdio.h>
#define MAX 10    //MAX adalah nama variabel (bisa bebas) yang diberi nilai 10

void isiArray(int arr[]);
void showArray(int arr[]);
int jumlahAllElemen(int arr[]);
int cariNilaiTerbesar(int arr[]);
void main()
{
	int array[MAX], sum, terbesar, pilihan;
	do{
		system("cls");
		puts("Menu Hitung Persegi");
		puts("1. Isi Array");
		puts("2. Tampilkan Isi Array");
		puts("3. Tampilkan Hasil Penjumlahan Semua Elemen");
		puts("4. Tampilkan Nilai Terbesar");
		puts("0. Exit");
		printf("Pilih menu: ");scanf("%d",&pilihan);
		switch(pilihan)
		{
			case 1:	isiArray(array);
				getch();
				break;
```

```
                  case 2:     showArray(array);
                              getch();
                              break;
                  case 3:     sum=jumlahAllElemen(array);
                              printf("Total Semua Elemen: %d",sum);
                              getch();
                              break;
                  case 4:     terbesar=cariNilaiTerbesar(array);
                        printf("Bilangan Terbesar: %d",terbesar);
                              getch();
                              break;
                  case 0:     break;
                  default:    printf("Maaf menu tidak ditemukan!");
                              getch();
                              break;
            }
      }while(pilihan!=0);
}
void showArray(int arr[])
{
      int i;
      for(i=0;i<MAX;i++){
            printf("%d\n",arr[i]);
      }
}
void isiArray(int arr[])
{
      int i;
      for(i=0;i<MAX;i++){
            printf("Masukkan Nilai Indeks ke %d: ",i);scanf("%d",&arr[i]);
      }
}
int jumlahAllElemen(int arr[])
```

```
{
	int i, total=0;
	for(i=0;i<MAX;i++){
		total=total+arr[i];
	}
	return total;
}
int cariNilaiTerbesar(int arr[])
{
	int i, bil_besar=arr[0];
	for(i=0;i<MAX;i++){
		if(arr[i]>bil_besar){
			bil_besar=arr[i];
		}
	}
	return bil_besar;
}
```

Guided 2

```
#include <stdio.h>
#define MAX 5

void isiArray(int arr[]);
void showArray(int arr[]);
void bubbleSortArray(int arr[]);

void main()
{
	int array[MAX], pilihan;
	do{
		system("cls");
		puts("Menu Hitung Persegi");
		puts("1. Isi Array");
		puts("2. Tampilkan Isi Array");
		puts("3. Urutkan Ascending Array");
		puts("0. Exit");
```

```
            printf("Pilih menu: ");scanf("%d",&pilihan);
            switch(pilihan)
            {
                  case 1:     isiArray(array);
                              getch();
                              break;
                  case 2:     showArray(array);
                              getch();
                              break;
                  case 3:     bubbleSortArray(array);
                              getch();
                              break;
                  case 0:     break;
                  default:    printf("Maaf menu tidak ditemukan!");
                              getch();
                              break;
            }
      }while(pilihan!=0);
}
void bubbleSortArray(int arr[])
{
      int i, j, temp;
      for(i=0;i<MAX;i++){
            for(j=i;j<MAX;j++){
                  if(arr[j]<arr[i]){
                        temp=arr[i];
                        arr[i]=arr[j];
                        arr[j]=temp;
                  }
            }
      }
}
void showArray(int arr[])
{
      int i;
      for(i=0;i<MAX;i++){
            printf("%d\n",arr[i]);
      }
}
```

```
void isiArray(int arr[])
{
	int i;
	for(i=0;i<MAX;i++){
		printf("Masukkan Nilai Indeks ke %d: ",i);scanf("%d",&arr[i]);
	}
}
```

Guided 3

```
#include <stdio.h>
#define MAX 3

void isiMatrix(int mat[MAX][MAX]);
void showMatriks(int mat[MAX][MAX]);
void sumMatriks(int mat_a[MAX][MAX], int mat_b[MAX][MAX], int total[MAX][MAX]);

void main()
{
	int a[MAX][MAX], b[MAX][MAX], mat_total[MAX][MAX],pilihan;
	do{
		system("cls");
		puts("Menu Hitung Persegi");
		puts("1. Isi Matriks A");
		puts("2. Isi Matriks B");
		puts("3. Tampilkan Isi Matriks A");
		puts("4. Tampilkan Isi Matriks B");
		puts("5. Jumlahkan Matriks");
		puts("0. Exit");
		printf("Pilih menu: ");scanf("%d",&pilihan);
		switch(pilihan)
		{
			case 1:	isiMatrix(a);
				getch();
				break;
			case 2:	isiMatrix(b);
				getch();
				break;
			case 3:	showMatriks(a);
				getch();
```

```
                              break;
                  case 4:     showMatriks(b);
                              getch();
                              break;
                  case 5:     sumMatriks(a,b,mat_total);
                              showMatriks(mat_total);
                              getch();
                              break;
                  case 0:     break;
                  default:    printf("Maaf menu tidak ditemukan!");
                              getch();
                              break;
            }
      }while(pilihan!=0);
}
void isiMatrix(int mat[MAX][MAX])
{
      int i, j;
      for(i=0;i<MAX;i++){
            for(j=0;j<MAX;j++){
                  printf("Input      Elemen      Baris      %d      Kolom      %d:
",i,j);scanf("%d",&mat[i][j]);
            }
      }
}
void showMatriks(int mat[MAX][MAX])
{
      int i, j;
      for(i=0;i<MAX;i++){
            for(j=0;j<MAX;j++){
                  printf("%d\t",mat[i][j]);
            }
            printf("\n");
      }
}
void sumMatriks(int mat_a[MAX][MAX], int mat_b[MAX][MAX], int total[MAX][MAX])
{
      int i, j;
      for(i=0;i<MAX;i++){
            for(j=0;j<MAX;j++){
```

```
                total[i][j]=mat_a[i][j]+mat_b[i][j];
            }
        }
}
```

PERTEMUAN V | RECORD

5.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 5, mahasiswa diharapkan dapat:

a. Memahami penggunaan Record dalam program.
b. Mengimplementasikan Array of record pada program.

5.2 Materi Praktikum

5.2.1 Record

Record disebut juga sebagai *struct.* Record adalah suatu tipe data yang terdiri dari gabungan 2 variabel atau lebih. Sebuah record memiliki beberapa atribut. Sebagai contoh, record yang akan disimpan adalah mobil. Maka *struct* untuk menyimpan data ini adalah:

```
typedef struct{
    int tahun;
    char merk[255];
}Mobil;
```

Mobil adalah sebuah tipe data baru yang memiliki atribut plat dan merk.

Struct juga dapat digunakan untuk membuat tipe data baru dari tipe data dasar dengan nama yang berbeda. Misalnya:

```
typedef char string[255];
typedef struct{
    int tahun;
    string merk;
}Mobil;
```

Akses atribut *struct* dapat dilakukan dengan menggunakan tanda titik (.) dari variabel yang bertipe data *struct* tersebut. Misalnya:

```
Mobil m;
m.plat=123;
strcpy(m.merk,"Toyota");
```

5.2.2 Array of Record

Penggunaan *struct* dapat dikombinasikan dengan array untuk menyimpan data. Data dapat disimpan sebanyak lebar array yang ditentukan. Contoh:

```
Mobil mbl[10];
```

Maka terdapat array mbl yang bertipe Mobil yang dapat digunakan untuk menyimpan 10 data mobil. Akses data dalam array of record dilakukan dengan menggunakan indeks array diikuti nama atribut dalam struct. Contoh:

```
Mobil mbl[10];
mbl[0].plat=123;
strcpy(mbl[0].merk,”Toyota”);
```

5.3 Guided

Guided 1

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

typedef char string[255];
typedef struct{
      int tahun;
      string merk;
}Mobil;
void init(Mobil *m);
void entryData(Mobil *m);
void show(Mobil m);

void main()
{
      int pilihan;
      Mobil m;
      do{
            system("cls");
            puts("Menu Hitung Persegi");
            puts("1. Inisialisasi Data");
            puts("2. Masukkan Data Mobil");
            puts("3. Tampilkan Data Mobil");
            puts("0. Exit");
            printf("Pilih menu: ");scanf("%d",&pilihan);
            switch(pilihan)
            {
                  case 1:     init(&m);
                              getch();
```

```
                        break;
                case 2:     entryData(&m);
                            getch();
                            break;
                case 3:     show(m);
                            getch();
                            break;
                case 0:     break;
                default:    printf("Maaf menu tidak ditemukan!");
                            getch();
                            break;
            }
      }while(pilihan!=0);
}
void init(Mobil *m)
{
      (*m).tahun=0;
      strcpy((*m).merk,"-");
}
void entryData(Mobil *m)
{
      printf("Input Tahun: ");scanf("%d",&(*m).tahun);
      //printf("Input Merk: ");scanf("%s",&(*m).merk);
      fflush(stdin);
      printf("Input Merk: ");gets((*m).merk);

}
void show(Mobil m)
{
      puts("Data Mobil");
      printf("Tahun: %d\n",m.tahun);
      printf("Merk: %s",m.merk);
}
```

Guided 2

```
#include <stdio.h>
#include <string.h>
#define MAX 3
```

```
typedef char string[255];
typedef struct{
      int tahun;
      string merk;
}Mobil;
void init(Mobil m[]);
void entryData(Mobil m[]);
void show(Mobil m[]);

void main()
{
      int pilihan;
      Mobil m[3];
      do{
            system("cls");
            puts("Menu Hitung Persegi");
            puts("1. Inisialisasi Data");
            puts("2. Masukkan Data Mobil");
            puts("3. Tampilkan Data Mobil");
            puts("0. Exit");
            printf("Pilih menu: ");scanf("%d",&pilihan);
            switch(pilihan)
            {
                  case 1:     init(m);
                              getch();
                              break;
                  case 2:     entryData(m);
                              getch();
                              break;
                  case 3:     show(m);
                              getch();
                              break;
                  case 0:     break;
                  default:    printf("Maaf menu tidak ditemukan!");
                              getch();
                              break;
            }
      }while(pilihan!=0);
}
void init(Mobil m[])
```

```
{
	int i;
	for(i=0;i<MAX;i++){
		m[i].tahun=0;
		strcpy(m[i].merk,"-");
	}
}
void entryData(Mobil m[])
{
	int i;
	for(i=0;i<MAX;i++){
		printf("Data Mobil Ke-%d\n",i+1);
		printf("Input Tahun: ");scanf("%d",&m[i].tahun);
		fflush(stdin);
		printf("Input Merk: ");gets(m[i].merk);
	}
}
void show(Mobil m[])
{
	int i;
	for(i=0;i<MAX;i++){
		printf("\nData Mobil Ke-%d",i+1);
		printf("\nTahun: %d",m[i].tahun);
		printf("\nMerk: %s",m[i].merk);
	}
}
```

PERTEMUAN VI | ABSTRACT DATA TYPE

6.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 6, mahasiswa diharapkan dapat:

a. Melakukan abstraksi data dari suatu permasalahan yang ada.
b. Menganalisa atribut dan operasi yang diperlukan dari sebuah ADT
c. Membuat program yang mengimplementasikan ADT.

6.2 Guided

Sebagai bentuk pelayanan kesehatan yang lebih baik terhadap pasien, rumah sakit “Cepat Sembuh” memberikan beberapa layanan kesehatan di antaranya adalah:

a. Medical Check-up

   Layanan medical check-up digunakan untuk mengecek kesehatan fisik pasien. Data dari pasien yang dihasilkan dari layanan ini adalah tinggi badan, berat badan, umur, dan tekanan darah pasien (systole/diastole). Medical Check-up ini selalu dilakukan setiap pasien memeriksakan diri di rumah sakit.

b. Transfusi Darah

   Layanan ini diberikan kepada pasien dalam keadaan kritis yang memerlukan darah. Data yang diperlukan adalah golongan darah pasien.

c. Dokter Pribadi

   Setiap pasien di rumah sakit akan diberi 1 dokter khusus sebagai dokter pribadi pasien. Maka data pasien yang dihasilkan nantinya memiliki nama dokter masing-masing.

d. Riwayat Pasien

   Riwayat pasien adalah riwayat yang berisi data pasien saat memeriksakan diri di rumah sakit. Data riwayat berisi hasil medical checkup ditambah dengan ID Riwayat, tanggal periksa, keluhan pasien, catatan dokter, dan resep.

e. Member Rumah Sakit

   Layanan member rumah sakit ini diberikan kepada pasien agar proses pencatatan data pasien menjadi lebih rapi dan lebih mudah. Untuk pembuatan member data yang diperlukan dari

pasien adalah: ID pasien, nomor ktp, nama, jenis kelamin, tanggal lahir, alamat, nomor telepon, nomor bpjs/askes, dan tanggal pendaftaran.

Buatlah program yang dapat digunakan untuk mengakomodasi keperluan rumah sakit tersebut agar dapat:

1. Melakukan pengelolaan data pasien:
    a. Insert data pasien
    b. Update data pasien by ID pasien
    c. Delete data pasien by ID pasien
2. Melakukan pengelolaan riwayat pasien:
    a. Insert riwayat by ID pasien
    b. Update riwayat by ID pasien & ID riwayat
    c. Delete riwayat pasien berdasarkan ID pasien
3. Menampilkan data member pasien (seperti tampilan di bawah)

Catatan:

- Saat ini, program tidak menggunakan array (karena hanya untuk 1 data saja).
- Pastikan tampilan untuk poin 3 dan 4 menampilkan data yang lengkap.

1. Contoh display data pasien:

Kartu Kesehatan Pasien
Nomor ID Pasien: 2017110001 Nama: Vena Ananda

PERTEMUAN VII | ARRAY OF ADT

7.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 7, mahasiswa diharapkan dapat:

a. Memahami implementasi array of ADT.
b. Memahami penggunaan array of ADT dalam program.

7.2 Guided

Modifikasi program yang sudah anda buat di pertemuan VI dengan menggunakan array! Buatlah agar program tersebut dapat:

1. Melakukan pengelolaan data pasien:
    a. Insert 5 data pasien:
        i. insert hanya boleh dilakukan satu per satu, bukan 5 data sekaligus.
        ii. ID pasien harus unik, jika sudah pernah dimasukkan sebelumnya maka sistem akan meminta input ulang ID pasien
    b. Update data pasien by ID pasien
    c. Delete data pasien by ID pasien
        i. Delete data pasien juga sekaligus delete riwayat pasien
2. Melakukan pengelolaan riwayat pasien:
    a. Insert riwayat (maksimal 3) by ID pasien
        i. Insert hanya boleh dilakukan satu per satu, bukan 3 data sekaligus
        ii. ID riwayat harus unik, jika sudah pernah dimasukkan sebelumnya maka sistem akan meminta input ulang ID Riwayat
    b. Update riwayat by ID pasien & ID riwayat
    c. Delete riwayat terakhir pasien berdasarkan ID pasien
3. Menampilkan data pasien
4. Menampilkan data riwayat pasien

PERTEMUAN VIII | STACK

8.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 8, mahasiswa diharapkan dapat:

a. Memahami konsep stack.
b. Mengimplementasikan konsep stack dalam sebuah kasus.

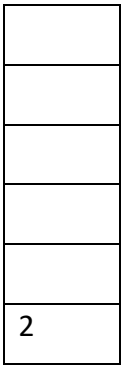
8.2 Guided

1. Buatlah stack yang dapat digunakan menyimpan bilangan integer sebanyak 6 bilangan. Lalu buatlah operasi menghitung bilangan yang dapat melakukan penjumlahan, pengurangan, perkalian dan pembagian secara otomatis dengan ketentuan sebagai berikut:
    a. Pada prinsipnya saat operasi perhitungan dilakukan semua bilangan dikeluarkan dari stack (pop).
    b. Jika stack kosong atau hanya terisi 1 bilangan maka tidak ada operasi yang dapat dilakukan
    c. Jika stack berisi 2 bilangan maka operasi pembagian akan dilakukan dan hasilnya dimasukkan lagi ke dalam stack.
    d. Jikat stack berisi lebih dari 2 bilangan dan tidak penuh, maka operasi pengurangan akan dilakukan dan hasilnya dimasukkan lagi ke stack.
    e. Jika stack penuh dan semua bilangan di dalam stack genap, maka operasi perkalian akan dilakukan dan hasilnya dimasukkan lagi ke dalam stack.
    f. Jika stack penuh dan semua bilangan di dalam stack ganjil, maka operasi penjumlahan akan dilakukan dan hasilnya dimasukkan lagi ke dalam stack.
    g. Jika ketentuan di a-f tidak terpenuhi maka tidak ada operasi yang dilakukan.

    Program hanya memiliki 3 menu:
    1. Push Bilangan
    2. Pop Bilangan
    3. Hitung Bilangan

Ilustrasi:

Stack kosong atau hanya 1 bilangan: Operasi hitung tidak dapat dilakukan



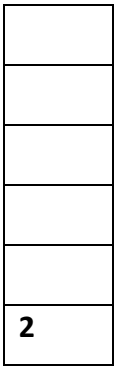
Stack berisi 2 bilangan: Operasi hitung pembagian

Pop 1: 5

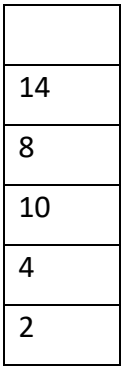
Pop 2: 2

Maka 5:2 = 2

2 dimasukkan kembali ke stack

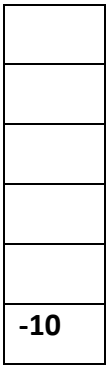


Stack berisi 5 bilangan: Operasi hitung pengurangan

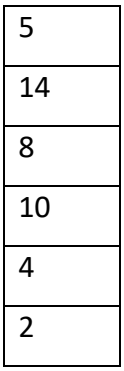


Maka 14-8-10-4-2 = -10

-10 dimasukkan kembali ke stack

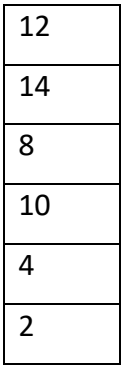


Stack terisi penuh: Tidak ada operasi hitung



Karena tidak ditemukan semua bilangan genap atau semua bilangan ganjil, maka tidak ada operasi hitung.

Stack terisi penuh bilangan genap: Operasi perkalian



Semua bilangan genap, maka operasi perkalian dilakukan:

12x14x8x10x4x2 = 107520

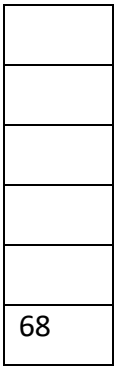
107520 dimasukkan kembali ke dalam stack.

Stack terisi penuh bilangan ganjil: Operasi pengurangan

Semua bilangan ganjil, maka operasi penjumlahan dilakukan:

37+1+9+13+3+5 = 68

68 dimasukkan kembali ke dalam stack.



PERTEMUAN IX | QUEUE

9.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 9, mahasiswa diharapkan dapat:

a. Memahami konsep Queue.
b. Mengimplementasikan Queue dalam sebuah kasus.

9.2 Guided

1. Sebagai solusi untuk menghadapi masalah melonjaknya jumlah wisatawan di pantai Biru, pemerintah daerah melakukan pemisahan tempat parkir untuk mobil dan motor. Kapasitas untuk tempat parkir mobil adalah 10 unit dan motor 10 unit. Biaya parkir yang harus dibayarkan untuk mobil adalah 3000/jam dan motor 2000/jam. Informasi yang disimpan untuk setiap unit adalah nomor polisi dan jam masuk. Buatlah program dengan konsep Queue dengan ketentuan:
    a. Buat ADT mobil, motor, waktu, dan Queue!
    b. Terdapat menu:
        a. Kendaraan datang
        b. Kendaraan keluar
        c. Tampilkan Total Kendaraan (Mobil & Motor)
    c. Pada saat Kendaraan datang, jika kendaraan yang dimasukkan adalah mobil maka dimasukkan ke Queue mobil, jika motor dimasukkan ke dalam queue motor. Waktu masuk juga perlu dimasukkan.
    d. Pada saat kendaraan keluar, jika kendaraan yang dikeluarkan adalah motor, maka dequeuer dari queue motor, jika mobil maka dequeue dari queue mobil. Waktu keluar juga dimasukkan dan dihitung biaya totalnya lalu ditampilkan (tidak disimpan).
    e. Tampilkan total kendaraan untuk menampilkan jumlah kendaraan saat ini baik mobil maupun motor.

PERTEMUAN X | POINTER

10.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 10, mahasiswa diharapkan dapat:

a. Memahami konsep pointer dalam program.
b. Memahami peran pointer dalam program.
c. Memahami cara menggunakan dan mengakses pointer.

10.2 Guided

1. Buatlah sebuah fungsi yang mengembalikan nilai total dari sebuah array of integer. Gunakan pointer yang menunjuk ke array tersebut untuk menghitung total penjumlahan semua bilangan di dalam array!
2. Buatlah sebuah program untuk menyimpan data mahasiswa. Atribut yang disimpan adalah: nama, NIM, nilai UTS, nilai UAS, dan nilai Tugas. Gunakan konsep **array dengan pointer** untuk membuat program. Data mahasiswa yang dapat disimpan maksimal adalah 10 mahasiswa.

   Gunakan Kode di bawah ini untuk ADT nya:

```
typedef struct
{
   char Nama[25];
   char NIM[10];
   int UTS;
   int UAS;
   int Tugas;
}Mahasiswa;


typedef Mahasiswa* addressMhs; //pointernya
```

   Program memiliki menu untuk input data, tampil data, hapus data mahasiswa by NIM.

   Catatan:

   - Proses input, edit, dan hapus data tidak boleh dilakukan dengan cara akses menggunakan titik (dot)
   - Pengaksesan data yang diperbolehkan hanya menggunakan pointer, misalnya menggunakan pointer dengan nama addressMhs. Contoh penggunaan addressMhs:

```
void main()
{
   Mahasiswa M[10];
   addressMhs PMhs;
   PMhs=&M[0]; // pointer menunjuk elemen pertama dari array
   scanf("%s",PMhs->Nama); // mengisi elemen 0 array field Nama
   PMhs++; //pointer sekarang menunjuk ke elemen kedua
   PMhs=&M[0]; //pointer sekarang menunjuk ke elemen pertama lagi
}
```

PERTEMUAN XI | LINEAR LINKED LIST

11.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 11, mahasiswa diharapkan dapat:

a. Memahami konsep linear linked list.
b. Memahami perbedaan antara array dengan linear linked list.
c. Mengimplementasikan penggunaan linked list dalam sebuah kasus.
d. Memahami operasi-operasi linear linked list.

11.2 Guided

1. Sebuah toko membutuhkan program yang diperlukan untuk mendata barang yang dijual. Setiap barang memiliki kode barang, nama, jumlah, harga, dan tanggal beli. Pemilik toko menginginkan banyaknya barang yang dapat dimasukkan ke program tak terbatas.

   Buatlah program dengan menggunakan konsep linear linked list dengan menu:

   a. Insert Barang (menggunakan Insert First)
   b. Hapus Barang (menggunakan Delete First)
   c. Tampilkan Semua Barang
   d. Insert Barang Setelah Kode Barang Tertentu (menggunakan Insert After)
   e. Hapus Barang By Kode Barang Tertentu
   f. Insert Barang di Urutan Terakhir (menggunakan Insert Last)
   g. Hapus Barang di Urutan Terakhir (menggunakan Delete Last)

PERTEMUAN XII | CIRCULAR LINKED LIST

12.1 Tujuan Pembelajaran

Melalui materi yang disampaikan pada pertemuan 12, mahasiswa diharapkan dapat:

a. Memahami konsep circular linked list.
b. Memahami perbedaan antara linear linked list dengan circular linked list.
c. Mengimplementasikan penggunaan circular linked list dalam sebuah kasus.
d. Memahami operasi-operasi circular linked list.

12.2 Guided

1. Buatlah sebuah program yang dapat menyimpan data mahasiswa. Data yang disimpan adalah: nama, nim, ipk, dan jenis kelamin.

   Buatlah program dengan menggunakan konsep circular linked list dengan menu:

   a. Insert Mahasiswa (menggunakan Insert First)
   b. Hapus Mahasiswa (menggunakan Delete First)
   c. Tampilkan Semua Mahasiswa
   d. Insert Mahasiswa Setelah Nama Tertentu (menggunakan Insert After)
   e. Hapus Mahasiswa By NIM Tertentu
   f. Insert Mahasiswa di Urutan Terakhir (menggunakan Insert Last)
   g. Hapus Mahasiswa di Urutan Terakhir (menggunakan Delete Last)