
"Saya menyatakan bahwa saya malaksanakan ujian ini dengan jujur. Apabila nanti ditemukan kecurangan dalam pelaksanaan ujian ini, saya bersedia menerima sanksi yang diberikan."

Nama : Fadiah Mumtaz Andevi

NIM : 18320009

---

1. Representasi integer 16 digit bit
   a. Unsigned
      tertinggi: $2^{16} - 1 = 65536 - 1 = 65535$
      terendah: 0

      2's component
      tertinggi: $2^{16-1} - 1 = 12^{15} - 1 = 32768 - 1 = 32767$
      terendah: $-(2^{16-1}) = -(2^{15}) = -32768$

   b. Decimal: 83009
      Binary: 10100010001000001

   c. Least significant byte = 1 = A

   d. Tabel

x	y	operasi	binary	hexa	unsigned	Signed2C
1010 0101	1110 1111	(A & y) ^x	1010 0100	A4	164	164
1101 1011	1111 0011	(~(y>>4) && x) + A	1101 10101	DA	218	218
0010 1111	1010 0101	((x>>2) | y) *A	0010 1110	2E	46	46

   e. Program Dalam Bahasa C

```c
#include <stdio.h>


void bin(short int n)
{
    unsigned short int i;
    for (i = 1 << 7; i > 0; i = i / 2)
```

```
        (n & i) ? printf("1") : printf("0");
    printf("\n");
}

int main() {
    int NIM = 18320009;
    NIM = ((NIM / 100000) % 100) * 1000 + NIM % 1000;
    short int A = NIM & 1;
    printf ("Nilai A yaitu: %d \n", A);

    short int y = 0b11101111;
    short int x = 0b10100101;
    short int hasil = (A & y) ^ x;

    // printf
    printf("Binary : ");
    bin(hasil);

    printf("Hex : %x\n", hasil);
    printf("Unsigned : %u\n", hasil);
    printf("Signed 2C : %d\n", hasil);
    printf("\n");

    y = 0b11110011;
    x = 0b11011011;
    hasil = (A & y) ^ x;

    printf("Binary : ");
    bin(hasil);

    printf("Hex : %x\n", hasil);
    printf("Unsigned : %u\n", hasil);
    printf("Signed 2C : %d\n", hasil);
    printf(" \n");

    y = 0b10100101;
    x = 0b00101111;
    hasil = (A & y) ^ x;
```

```
    printf("Binary : ");
    bin(hasil);

    printf("Hex : %x\n", hasil);
    printf("Unsigned : %u\n", hasil);
    printf("Signed 2C : %d\n", hasil);

    return 0 ;
}
```

f. Berdasarkan perhitungan table dan program telah sama.

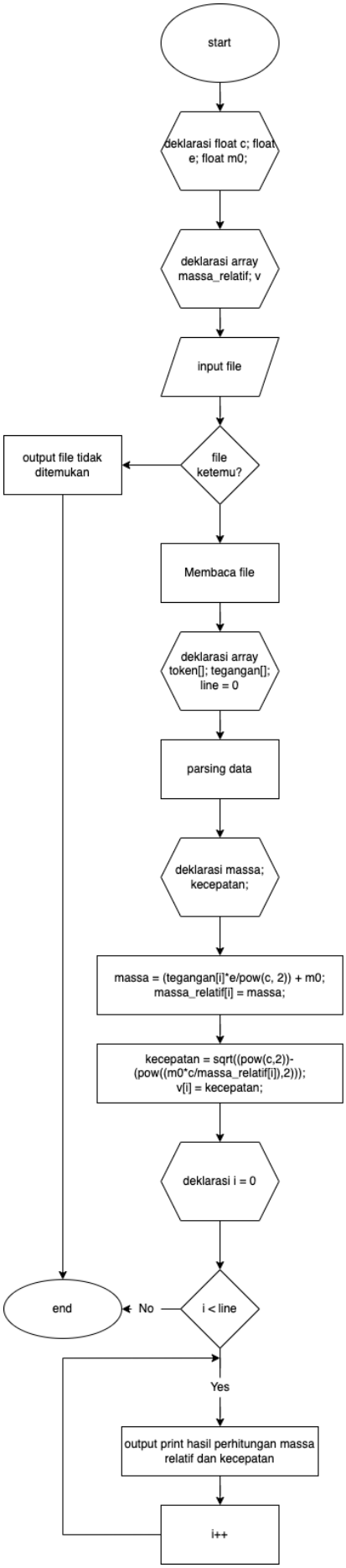
2. Massa Relatif dan Kecepatan Elektron

a. Pemrograman

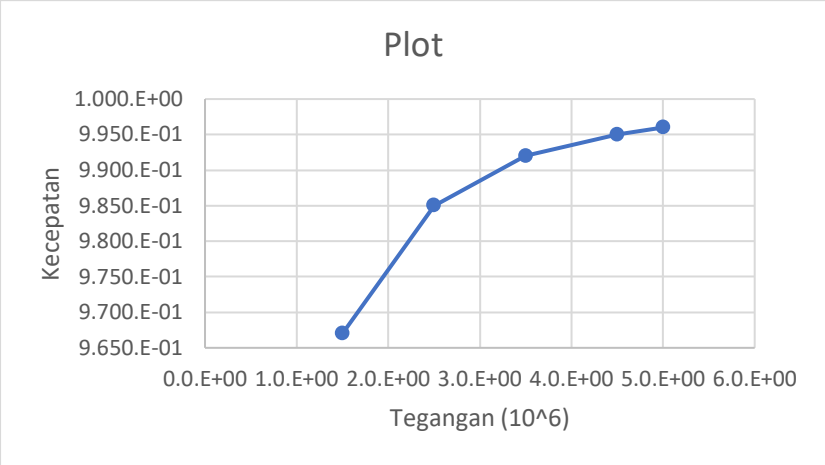
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#define MAX_LEN 255

int main(){
    //Kecepatan Cahaya (*10^8)
    float c = 2.9979;
    //Muatan Elektron (*10^-30)
    float e = 16.02;
    //Massa Diam (*10^-18)
    float m0 = 0.9109;

    //Array untuk Massa Relatif
    float massa_relatif[MAX_LEN];
    //Array untuk Kecepatan
    float v[MAX_LEN];

    char filename[MAX_LEN];
    printf("Masukkan file external: ");
    scanf("%s", filename);

    //open file
    FILE *fp;
```

```
    fp = fopen(filename, "r");
    if (fp==NULL){
        printf("File tidak ditemukan.");

    }
    else{
        float token[MAX_LEN];
        float tegangan[MAX_LEN];

        //input data ke array
        int line = 0;
        while(fgets(token, MAX_LEN, fp)){
            tegangan[line] = atof(token);
            line++;
        }

        //hitung massa relativistik dan kecepatan elektron
        float massa;
        float kecepatan;
        for(int i=0; i<line; i++){
            //Menghitung Massa (*10^-30)
            massa = (tegangan[i]*e/pow(c, 2)) + m0;
            massa_relatif[i] = massa;

            //Menghitung Kecepatan (10^8)
            kecepatan = sqrt((pow(c,2))-(pow((m0*c/massa_relatif[i]),2)));
            v[i] = kecepatan;
        }

        for(int i=0; i<line; i++){
            printf("Tegangan %f memiliki massa relatif %f x 10^-30 dan kecepatan %f x10^8.\n", tegangan[i],
massa_relatif[i], v[i]);
        }

        return 0;
    }
```

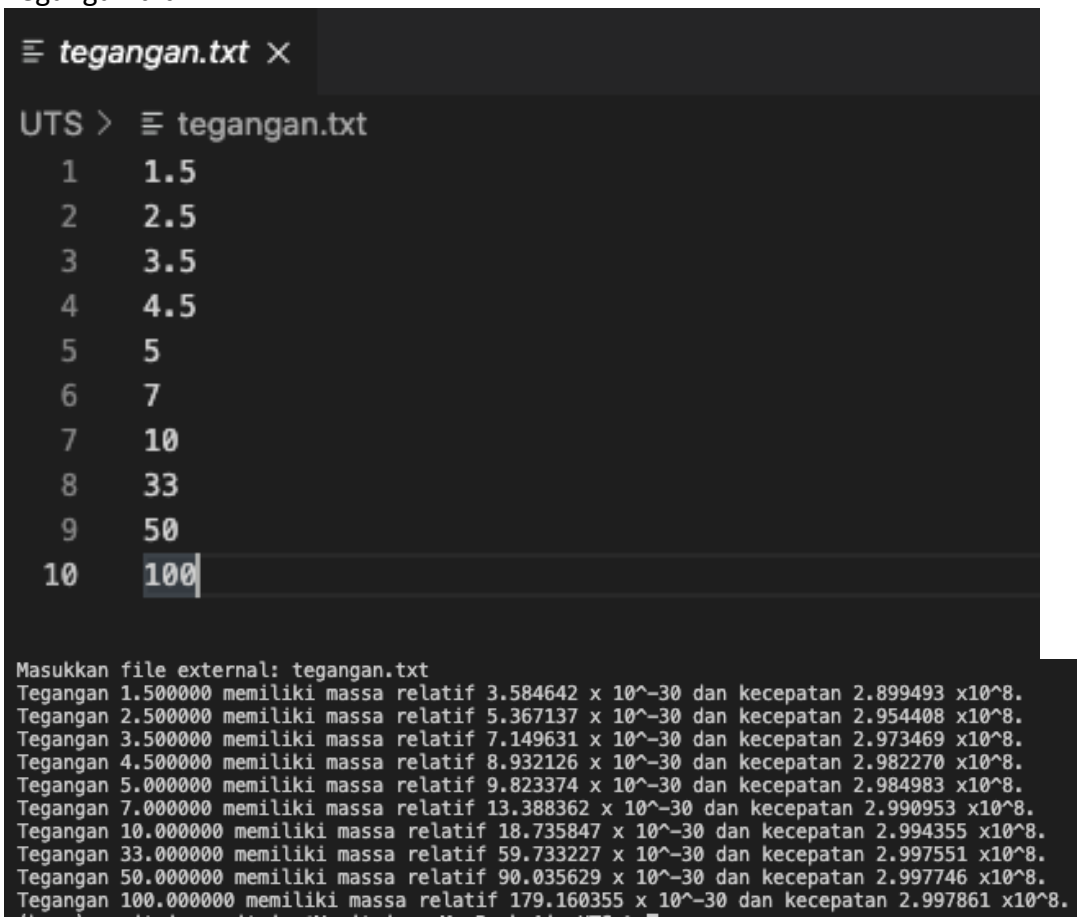
Flowchart


start
deklarasi float c; float e; float m0;
deklarasi array massa_relatif; v
input file
file ketemu?
output file tidak ditemukan
Membaca file
deklarasi array token[]; tegangan[]; line = 0
parsing data
deklarasi massa; kecepatan;
massa = (tegangan[i]*e/pow(c, 2)) + m0;
massa_relatif[i] = massa;
kecepatan = sqrt((pow(c,2))-(pow((m0*c/massa_relatif[i]),2)));
v[i] = kecepatan;
deklarasi i = 0
i < line
No
end
Yes
output print hasil perhitungan massa relatif dan kecepatan
i++

b. Plot



c. Tegangan.txt



3. Rangkaian Penapis
   a. Turunan

Persamaan Diferensial

$$RC \frac{d}{dt} V_o(t) + V_o(t) = V_{in}$$

$V_o(t) = V_{o1}(t) + V_{o2}(t)$

$V_{o1}(t) = Ae^{\frac{t}{RC}}$

$V_{o2} = B$

Dari pers. Diferensial didapat

$$RC \frac{d}{dt} V_{o2}(t) + V_{o2}(t) = V_{in}$$

$$RC \frac{dB}{dt} + B = V_{in}$$

$$0 + B = V_{in}$$

$$B = V_{in}$$

$V_o(t) = Ae^{-\frac{t}{RC}} + B$

$= Ae^{-\frac{t}{RC}} + V_{in}$

$V_o(0) = A + V_{in}$

$A = V_o(0) - V_{in} = -V_{in}$

$V_o(t) = -V_{in} e^{-\frac{t}{RC}} + V_{in}$

$V_o(t) = V_{in}(1 - e^{-\frac{t}{RC}})$

$RC = \tau$ (tetap waktu)

$\tau = 20 \cdot 10^3 \times 183009 \times 10^{-12}$

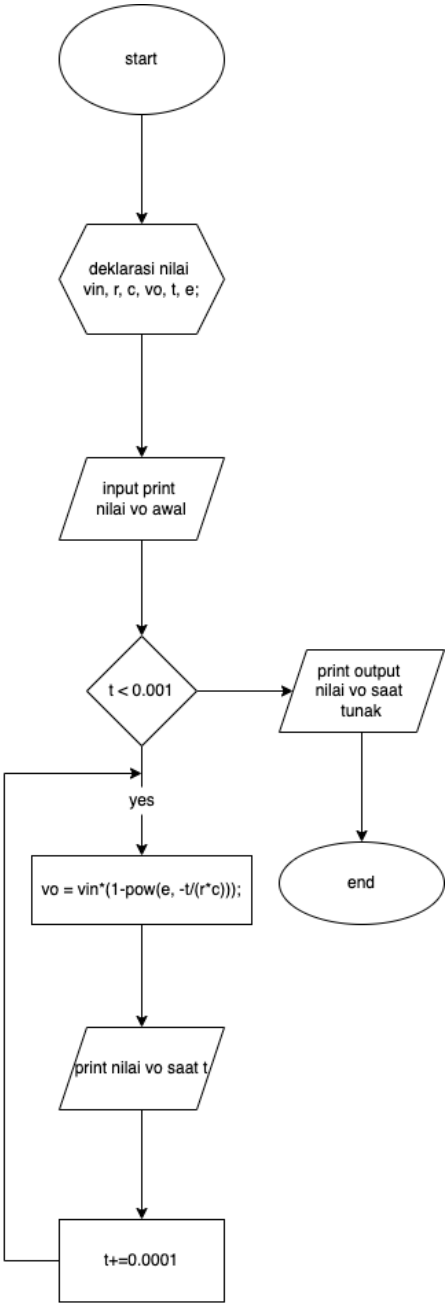
$= 3,660180 \times 10^{-3}$ s

$$V_o(t) = 5\left(1 - e^{-\frac{t}{3,66 \times 10^{-3}}}\right)$$

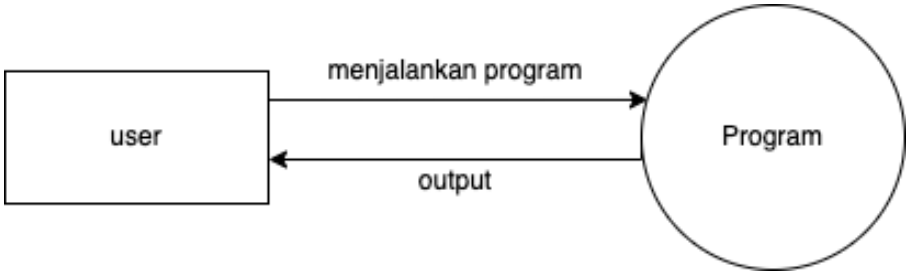
Maka

$$V_o = \begin{cases} 0, & t < 0 \\ 5\left(1 - e^{-t/3,66 \times 10^{-3}}\right), & t > 0 \end{cases}$$

b. Flowchart



c. DFD

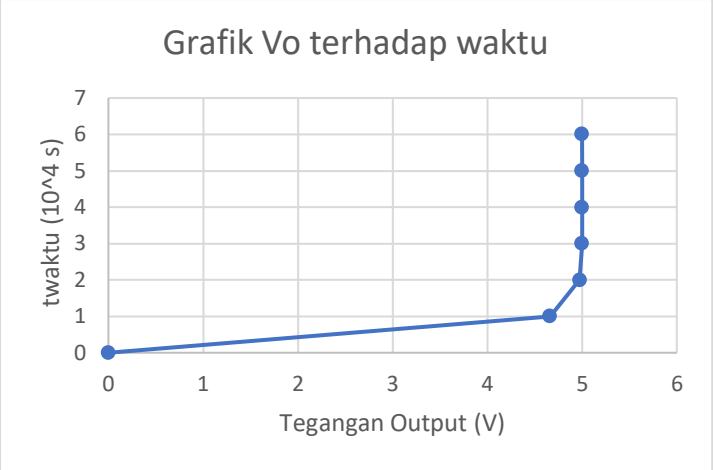


d. Program dalam Bahasa C

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

```c
int main (){

   //Nilai Vin
   float vin = 5.0;
   //Niai Resistansi
   float r = 20*10^(3);
   //Nilai Kapasitor
   float c = 183009*(pow(10,-12));
   //Nilai V output
   float vo = 0;
   //Waktu
   float t = 0;
   //nilai e
   float e = 2.718;

   printf("Vo saat t = %f yaitu %f\n", t, vo);

 //Vo tunak saat Vo bernilai 0
   for (float t=0; t < 0.001; t += 0.0001){
      vo = vin*(1-pow(e, -t/(r*c)));
      printf("Nilai Vo saat t = %f sebesar %f\n", t, vo);
   }

   printf("Vo tunak saat pertama Vo sama dengan Vin yaitu saat nilainya %f\n", vo);
   return 0;
}
```

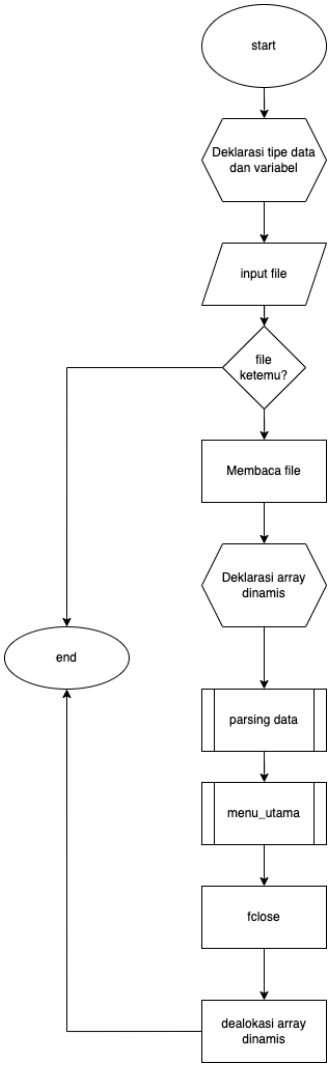
e. Grafik



f. Nilai konstanta
Ambil Ketika t = 0.0001 dimana apabila menggunakan rumus pada (a) didapati nilai sebesar 0.163 sedangkan pada program didapati nilai sebesar 0.161. Hasil tidak terlalu berbeda

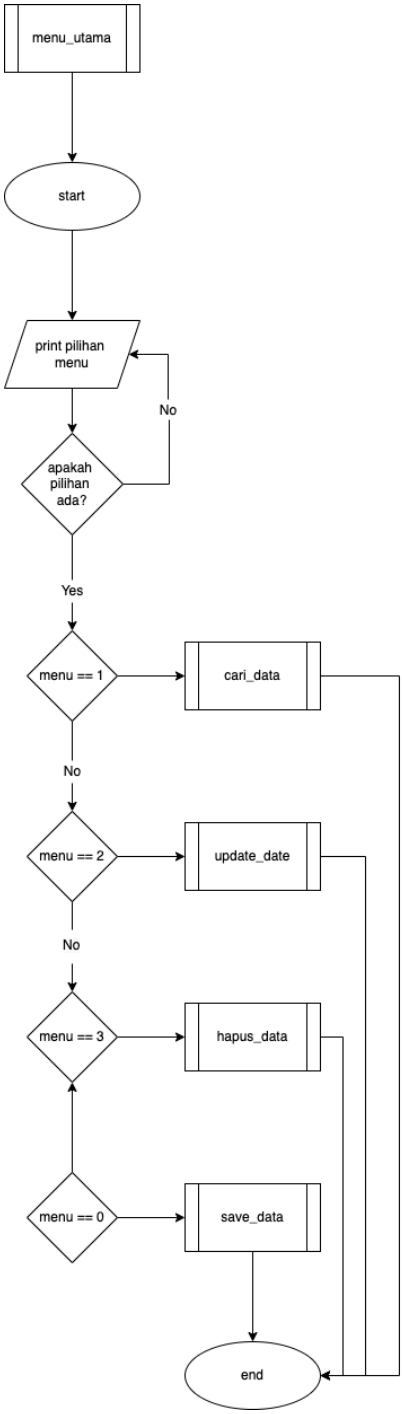
```
Vo saat t = 0.000000 yaitu 0.000000
Nilai Vo saat t = 0.000000 sebesar 0.000000
Nilai Vo saat t = 0.000100 sebesar 4.661088
Nilai Vo saat t = 0.000200 sebesar 4.977027
Nilai Vo saat t = 0.000300 sebesar 4.998443
Nilai Vo saat t = 0.000400 sebesar 4.999895
Nilai Vo saat t = 0.000500 sebesar 4.999993
Nilai Vo saat t = 0.000600 sebesar 5.000000
Nilai Vo saat t = 0.000700 sebesar 5.000000
Nilai Vo saat t = 0.000800 sebesar 5.000000
Nilai Vo saat t = 0.000900 sebesar 5.000000
Nilai Vo saat t = 0.001000 sebesar 5.000000
Vo tunak saat pertama Vo sama dengan Vin yaitu saat nilainya 5.000000
nilai konstanta waktu 161.503800
```

4. Pencatatan Data Penduduk
   a. Flowchart



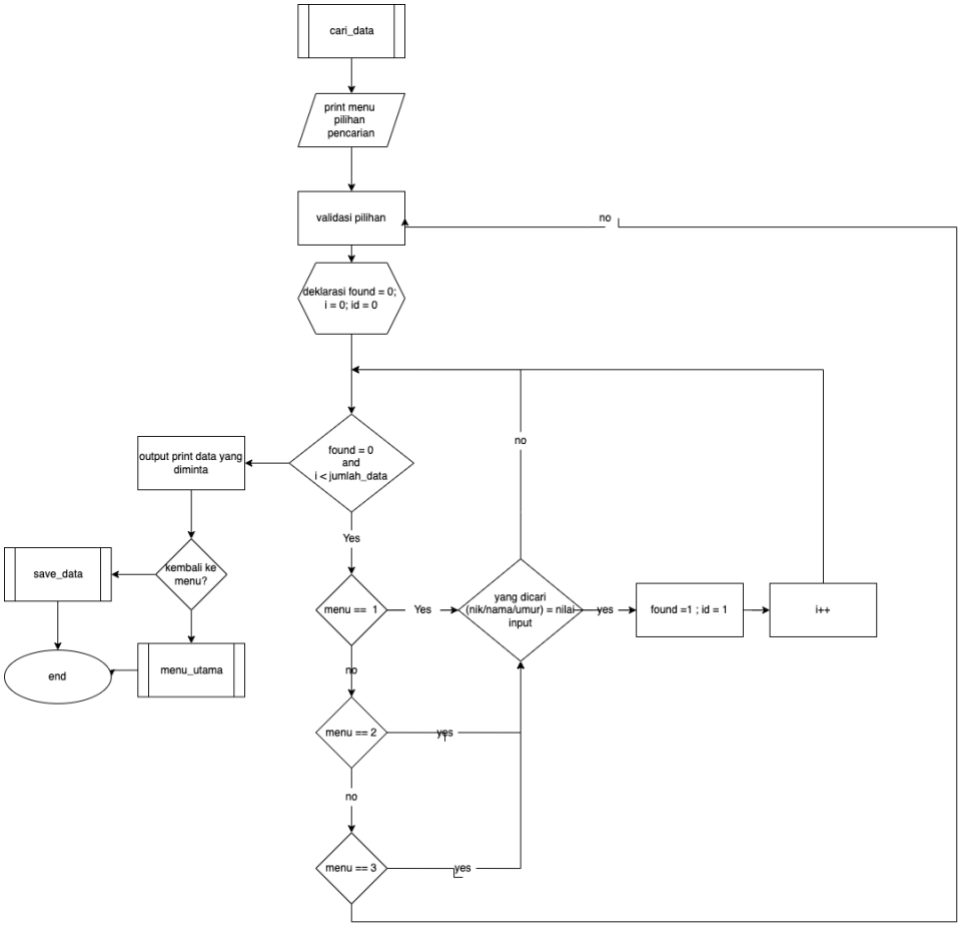
Gambar 4.1 Flowchart Main

Pertama yang dilakukan adalah deklarasi tipe data, lalu dilakukan input file untuk membaca file yang telah tesedia dimana apabila file invalid maka akan keluar output yang memberitahu bahwa file tidak ditemukan sementara apabila file ada maka program akan masuk ke fungsi parsing data, lalu ke menu utama, kemudian akan dilakukan penutupan file dan dealokasi array dinamis.



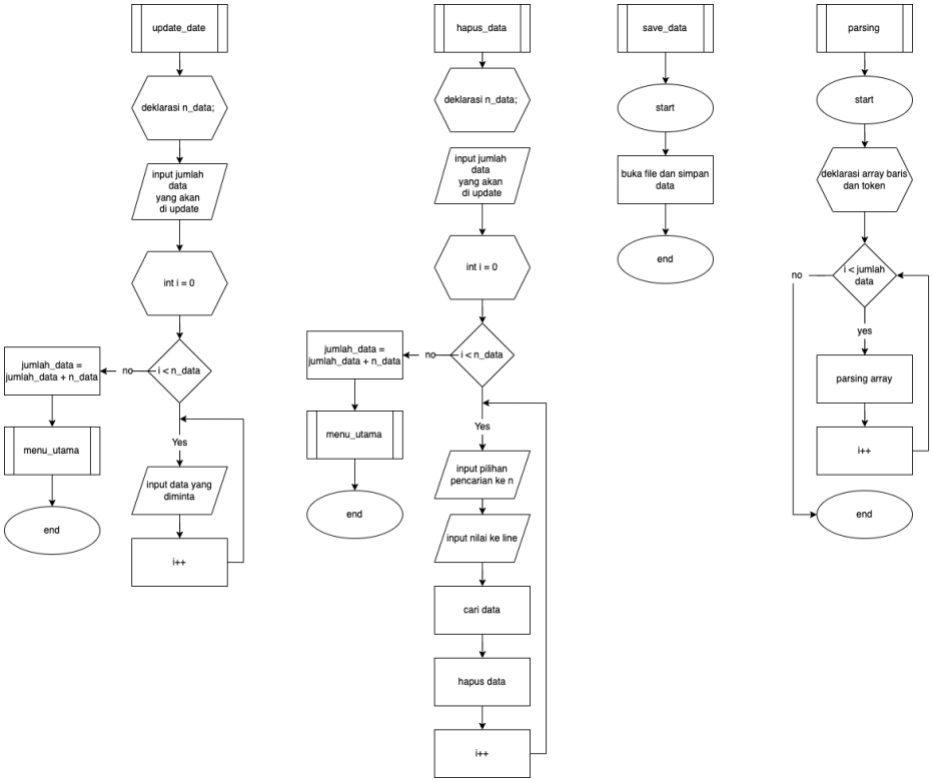
Gambar 4.2 Flowchart fungsi menu_utama

Pada fungsi ini akan membuat program menampilkan pilihan menu agar user dapat memilih. Apabila user memilih pilihan satu berarti user akan mencari data, pilihan menu 2 untuk mengupdate data, dan pilihan menu ketiga untuk menghapus data. Nantinya setelah melakukan fungsi tersebut, program akan menyimpan data diakhir.



Gambar 4.3 Flowchart fungsi cari_data

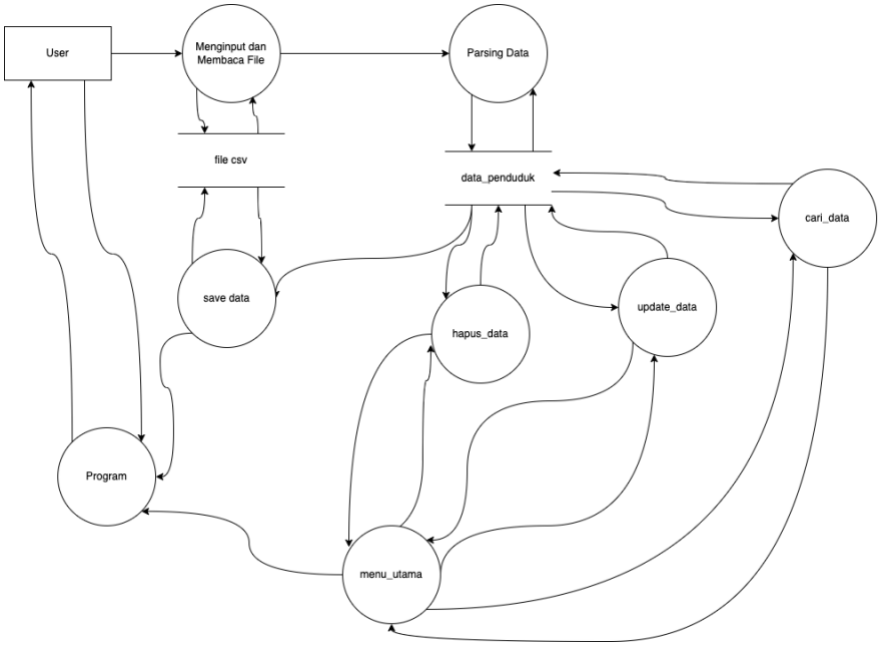
Pada fungsi ini, user akan diminta untuk memilih menu yaitu untuk pencarian data berdasarkan nik, nama, atau umur. Lalu kemudian pilihan menu akan divalidasi, apabila sesuai maka akan dilanjutkan. Menu 1 adalah apabila user melakukan pencarian berdasarkan nik, menu 2 nama, dan menu 3 berdasarkan umur. Apabila sudah tidak ada data yang dapat dicari lagi, maka program akan mengeluarkan keluaran berupa nama-nama yang dicari dan kemudian meminta apakah user masih mau mencari nama atau tidak.



Flowchart fungsi update_data, hapus_data, save_data, dan parsing

- Untuk update_data, pertama akan dideklarasi n_data yang kemudian user isi untuk input banyaknya data yang akan diupdate. Lalu dilakukan looping untuk mgninput data yang diminta. Hasilnya yaitu jumlah_data akan berubah dan program selesai
- Untuk hapus_data, sama seperti update_data dilakukan dulu deklarasi n_data untuk menyuruh user menginput banyaknya data yang akan dihapus, kemudian dilakukan looping untuk memilih data yang akan dihapus. Setelah semua data yang user mau dihapus, program akan mengubah jumlah_data dan akhirnya selesai.
- Save data hanya digunakan untuk menyimpan data yang telah diubah sesuai keinginann user
- Untuk parsing pertama akan dilakukan deklarasi array baris dan token lalu dilakukan looping untuk memisahkan data sesuai dengan kolomnya. Apabila semua kolom telah diparsing, maka program fungsi ini akan selsai

b. DFD



c. Program Bahasa C

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

#define MAX_LEN 255
#define column_size 8

typedef struct penduduk {
    int nik;
    char nama[MAX_LEN];
    char tempat_lahir[MAX_LEN];
    char tanggal_lahir[MAX_LEN];
    int umur;
    char gender[MAX_LEN];
    char goldar[MAX_LEN];
    char status[MAX_LEN];
    char pekerjaan[MAX_LEN];
} penduduk;

//Mencetak Menu saat Program Dibuka
void menu_utama(FILE * fp, penduduk * data_ktp, int jumlah_data);


//parsing data
```

```
void parsing(FILE * fp, penduduk * data, int jumlah_data){
    char line [MAX_LEN];
    char *token;
    for (int i= 0; i < jumlah_data; i++){
        fgets(line,MAX_LEN,fp);
        token = strtok(line,",");
        data[i].nik = atoi(token);

        for (int j = 0; j < column_size-1; j++){
            token = strtok(NULL,",");
            if (j == 0) {
                strcpy(data[i].nama,token);
            }
            else if (j == 1){
                char temp[MAX_LEN];
                strcpy(temp,token);
                token = strtok(NULL,",");
                strcat(temp,", ");
                strcat(temp, token);
                strcpy(data[i].tempat_lahir,temp);
            }
            else if (j == 2) {
                strcpy(data[i].tanggal_lahir,token);
            }
            else if (j == 3) {
                data[i].umur = atoi(token);
            }
            else if (j == 4) {
                strcpy(data[i].gender,token);
            }
            else if (j == 5) {
                strcpy(data[i].goldar,token);
            }
            else if (j == 6) {
                strcpy(data[i].status,token);
            }
            else if (j == 7) {
                strcpy(data[i].pekerjaan,token);
            }
```

```
        }
    }
}

//menyimpan data
void save_data(FILE *fp, penduduk * data_penduduk, int jumlah_data){
    FILE * ftemp;

    ftemp = fopen("temp.txt","w");

    for (int i=0; i < jumlah_data; i++){
        fprintf(ftemp,"%d,%s,%s,%s,%d,%s,%s,%s,%s\n",
        data_penduduk[i].nik, data_penduduk[i].nama, data_penduduk[i].tempat_lahir,
        data_penduduk[i].tanggal_lahir, data_penduduk[i].umur, data_penduduk[i].gender,
        data_penduduk[i].goldar, data_penduduk[i].status, data_penduduk[i].pekerjaan
        );

    }
    fclose(ftemp);
    rename("temp.txt","data.txt");
}

//Mencari data berdasarkan NIK/Nama/Umur
void cari_data (FILE * fp, penduduk * data_penduduk, int jumlah_data){
//input preferensi pencarian
    int menu;
    char line[MAX_LEN];
    printf("\nMenu Pencarian Data: \n");
    printf("1. Cari NIK\n");
    printf("2. Cari Nama\n");
    printf("3. Cari Umur\n");
    printf("\nData yang akan dicari: ");
    scanf("%d", &menu);

    //validasi input
    while (menu != 1 && menu != 2 && menu != 3){
        printf("Masukkan tidak valid\n");
        printf("\nPilihan: ");
        scanf("%d", &menu);
```

```
}

//input pencarian
if (menu == 1) {
    printf("\nMasukkan NIK: ");
}
else if (menu == 2) {
    printf("\nMasukkan Nama: ");
}
else { // menu==3
    printf("\nMasukkan Umur: ");
}
scanf("%s",line);


//pencarian
int found = 0;
int i = 0;
int id = 0;

while (found == 0 && i < jumlah_data){
    if(menu == 1){
        if (data_penduduk[i].nik == atoi(line)){
            found = 1;
            id = i;
        }
    }
    else if (menu == 2) {
        if (strstr(data_penduduk[i].nama,line)){
            found = 1;
            id = i;
        }
    }
    else { //menu == 3}
        if (data_penduduk[i].umur == atoi(line)){
            found = 1;
            id = i;
        }
    }
    i++;
```

```
    }

    //cetak hasil pencarian
    printf("\nHasil Pencarian\n");
    printf("NIK : %d\n", data_penduduk[id].nik);
    printf("Nama : %s\n", data_penduduk[id].nama);
    printf("Tempat Lahir : %s\n", data_penduduk[id].tempat_lahir);
    printf("Tanggal Lahir : %s\n", data_penduduk[id].tanggal_lahir);
    printf("Umur : %d\n", data_penduduk[id].umur);
    printf("Jenis Kelamin : %s\n", data_penduduk[id].gender);
    printf("Golongan Darah : %s\n", data_penduduk[id].goldar);
    printf("Status : %s\n", data_penduduk[id].status);
    printf("Pekerjaan : %s\n", data_penduduk[id].pekerjaan);

    char op;
    printf("\n Kembali ke menu utama?(y/n)");
    scanf("%c",&op);

    while ((&op != "y"||&op != "Y") && (&op != "n"||&op != "N")) {
        printf("\ninput tidak valid.");
        printf("\n Kembali ke menu utama?(y/n)");
        scanf("%c",&op);
    }

    if (&op == "yes" || &op == "Y") {
        menu_utama(fp,data_penduduk,jumlah_data);
    }
    else {
        cari_data(fp, data_penduduk,jumlah_data);
    }
}


//Update Data
void update_data (FILE * fp, penduduk * data_penduduk, int jumlah_data) {
    int n_data;
    printf("\nMenu Rekam Data\n");
    printf("Jumlah data yang akan direkam: ");
    scanf("%d", &n_data);
    data_penduduk = (penduduk*) realloc(data_penduduk,(jumlah_data+n_data)*sizeof(penduduk));
```

```c
//perekaman data masing-masing komponen
printf("\nPerekaman Data");
for (int i = 0; i < n_data; i++){
    printf("\nInput data ke-%d", i+1);
    printf("\nNIK: ");
    scanf("%d", &data_penduduk[jumlah_data+i].nik);
    fflush(stdin);

    printf("Nama: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].nama);
    fflush(stdin);

    printf("Tempat Lahir: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].tempat_lahir);
    fflush(stdin);

    printf("Tanggal Lahir: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].tanggal_lahir);
    fflush(stdin);

    printf("Umur: ");
    scanf("%*d",data_penduduk[jumlah_data+i].umur);
    fflush(stdin);

    printf("Jenis Kelamin: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].gender);
    fflush(stdin);

    printf("Golongan Darah: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].goldar);
    fflush(stdin);

    printf("Status: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].status);
    fflush(stdin);

    printf("Pekerjaan: ");
    scanf("%[^\n]%*c",data_penduduk[jumlah_data+i].pekerjaan);
```

```
        fflush(stdin);
    }

    jumlah_data = jumlah_data + n_data;
    printf("\nPerekaman data selesai. Program akan kembali ke menu utama.\n");
    menu_utama(fp, data_penduduk,jumlah_data);
}

//Menghapus Data
void hapus_data(FILE * fp, penduduk * data_penduduk, int jumlah_data) {
    //pengaturan ulang ukuran array
    int n_data;
    printf("\nMenu Penghapusan Data\n");
    printf("Jumlah data yang ingin dihapus: ");
    scanf("%d", &n_data);
    fflush(stdin);

    for (int i = 0; i < n_data; i++){
        //input pencarian
        char line[MAX_LEN];
        printf("\nMasukkan NIK: ");
        scanf("%s",line);

        //pencarian
        int found = 0;
        int j = 0;
        int id = 0;

        while (found == 0 && j < jumlah_data){
            (data_penduduk[j].nik = atoi(line));
            found = 1;
            id = j;
            j++;
        }

        //penghapusan data
        penduduk *temp;
        temp = data_penduduk;
        data_penduduk = (penduduk*) realloc(data_penduduk,(jumlah_data-1)*sizeof(penduduk));
```

```
        for(int k = 0; k < (jumlah_data-1); k++){
            if (k >= id){
                data_penduduk[k].nik = temp[k+1].nik;
                strcpy(data_penduduk[k].nama, temp[k+1].nama);
                strcpy(data_penduduk[k].tempat_lahir, temp[k+1].tempat_lahir);
                strcpy(data_penduduk[k].tanggal_lahir, temp[k+1].tanggal_lahir);
                data_penduduk[k].umur = temp[k+1].umur;
                strcpy(data_penduduk[k].gender, temp[k+1].gender);
                strcpy(data_penduduk[k].goldar, temp[k+1].goldar);
                strcpy(data_penduduk[k].status, temp[k+1].status);
                strcpy(data_penduduk[k].pekerjaan, temp[k+1].pekerjaan);
            }
        }
    }
    jumlah_data = jumlah_data - n_data;
    printf("\nPenghapusan data selesai. Program akan kembali ke menu utama.\n");
    menu_utama(fp, data_penduduk, jumlah_data);
}

//Menampilkan menu utama
void menu_utama(FILE * fp, penduduk * data_penduduk, int jumlah_data){
    int menu;
    printf("\nMenu Pencatatan Data Kependudukan: \n");
    printf("1. Cari Data\n");
    printf("2. Update Data\n");
    printf("3. Hapus Data\n");
    printf("0. Keluar\n");
    printf("\nYang akan dilakukan: ");
    scanf("%d", &menu);

    //validasi input
    while (menu != 1 && menu!= 2 && menu!= 3 && menu!= 0){
        printf("Masukkan tidak valid\n");
        printf("\nPilihan: ");
        scanf("%d", &menu);
    }

    if (menu == 1){
        cari_data(fp, data_penduduk, jumlah_data);
```

```
    }
    else if (menu == 2) {
        update_data(fp, data_penduduk, jumlah_data);
    }
    else if (menu == 3) {
        hapus_data(fp, data_penduduk, jumlah_data);
    }
    else {
        save_data(fp, data_penduduk, jumlah_data);
    }
}

int main(){
    //membuka file
    char filename[MAX_LEN];
    printf("Masukkan nama file: ");
    scanf("%s", filename);
    FILE * fp = fopen(filename, "r");

    //file tidak ditemukan
    while (!fp) {
        printf("File tidak ditemukan!\n\n");
        printf("Masukkan nama file: ");
        scanf("%s", filename);
        fp = fopen(filename,"r");
    }

    char line[MAX_LEN];
    fgets(line,MAX_LEN,fp);
    char rt[MAX_LEN];
    strcpy(rt,line);
    printf("\n");
    printf("Sistem Pencatatan Data Kependudukan %s\n",rt);

    //inisialisasi array
    fgets(line,MAX_LEN, fp);
    int jumlah_data = atoi(line);
    penduduk * data_penduduk = (penduduk*) malloc(jumlah_data*sizeof(penduduk));
```

```
    //pengisian array
    parsing(fp, data_penduduk, jumlah_data);
    menu_utama(fp, data_penduduk, jumlah_data);
    fclose(fp);
    free(data_penduduk);
    return 0;
}
5.
```