CHRISTOPHER CHANDRA 18320033
UTS PMC

1a. unsigned: nilai maksimum 65535 dan minimum 0 karena nonnegative. Dengan 2's complement, nilai maksimum adalah 32767 sedangkan nilai minimum -32768

b. 83033 -> 10100010001011001

c. A = LSB = 01011001

d.

x	y	Operasi	Binary	Hexa	Unsigned	Signed 2C
1010 0101	1110 1111	(A&y)^x				
1101 1011	1111 0011	(~(y>>4)&&x)+A				
0010 1111	1010 0101	((x>>2)|y)*A				

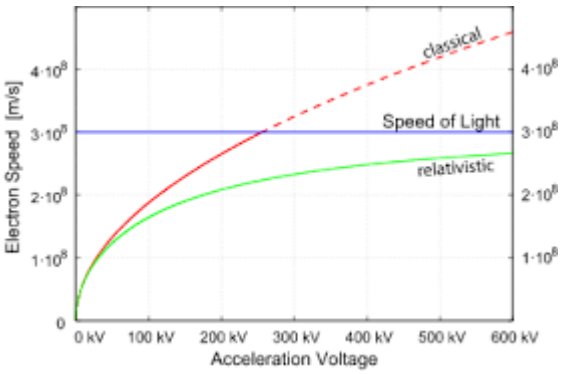
e.

f.

2a.

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define MAX_LEN 255
#define c 2.9979 * pow(10,8)
#define m0 9.109 * pow(10,-31)
#define e 1.602 * pow(10,-19)

float countmass(int V){
    float mass;
    mass = m0 + ((V*e*pow(10,6))/pow(c,2));
    return mass;
}
float countspeed(float mass){
    float speed;
    float l;
    l = 1- pow((m0/mass),2);
    speed = c*pow(l,0.5);
    return speed;
}
int main(){
    FILE* stream = fopen("tegangan.txt", "r");
    int V[10];
    float mass[10];
    int i = 0;
    while (fscanf(stream, "%d", &V[i]) != EOF){
        i++;
    }
    for (i=0; i<10;++i){
        mass[i] = countmass(V[i]);
        printf("Massa ke-[%d], %.2e\n",i+1, mass[i]);
    }
    printf("\n");
    for (i=0;i<10;++i){
        printf("Kecepatan ke-[%d], %.2e\n", i+1, countspeed(mass[i]));
    }
    fclose(stream);
    return 0;
}
```

b.



c.
Massa ke-[1], 3.57e-26
Massa ke-[2], 2.22e-24
Massa ke-[3], 1.56e-26
Massa ke-[4], 5.80e-23
Massa ke-[5], 4.30e-26
Massa ke-[6], 2.19e-24
Massa ke-[7], 6.36e-25
Massa ke-[8], 1.01e-21
Massa ke-[9], 6.15e-29
Massa ke-[10], 1.39e-26

Kecepatan ke-[1], 3.00e+08
Kecepatan ke-[2], 3.00e+08
Kecepatan ke-[3], 3.00e+08
Kecepatan ke-[4], 3.00e+08
Kecepatan ke-[5], 3.00e+08
Kecepatan ke-[6], 3.00e+08
Kecepatan ke-[7], 3.00e+08
Kecepatan ke-[8], 3.00e+08
Kecepatan ke-[9], 3.00e+08
Kecepatan ke-[10], 3.00e+08

Process returned 0 (0x0) execution time : 0.048 s
Press any key to continue.

3a.

We assume an initial v oltage V_0 on the capacitor , although this is not necessary for the step response. Since the v oltage of a capacitor cannot change instantaneously,

$$v(0^-) = v(0^+) = V_0 \tag{7.40}$$

where $v(0^-)$ is the voltage across the capacitor just before switching and $v(0^+)$ is its voltage immediately after switching. Applying KCL, we have

$$C\frac{dv}{dt} + \frac{v - V_s u(t)}{R} = 0$$

or

$$\frac{dv}{dt} + \frac{v}{RC} = \frac{V_s}{RC}u(t) \tag{7.41}$$

where v is the voltage across the capacitor. For $t > 0$, Eq. (7.41) becomes

$$\frac{dv}{dt} + \frac{v}{RC} = \frac{V_s}{RC} \tag{7.42}$$

Rearranging terms gives

$$\frac{dv}{dt} = -\frac{v - V_s}{RC}$$

or

$$\frac{dv}{v - V_s} = -\frac{dt}{RC} \tag{7.43}$$

Integrating both sides and introducing the initial conditions,

$$\ln(v - V_s)\Big|_{V_0}^{v(t)} = -\frac{t}{RC}\Big|_0^t$$

$$\ln(v(t) - V_s) - \ln(V_0 - V_s) = -\frac{t}{RC} + 0$$

or

$$\ln\frac{v - V_s}{V_0 - V_s} = -\frac{t}{RC} \tag{7.44}$$

Taking the exponential of both sides

$$\frac{v - V_s}{V_0 - V_s} = e^{-t/\tau}, \qquad \tau = RC$$

$$v - V_s = (V_0 - V_s)e^{-t/\tau}$$

or

$$v(t) = V_s + (V_0 - V_s)e^{-t/\tau}, \qquad t > 0 \tag{7.45}$$

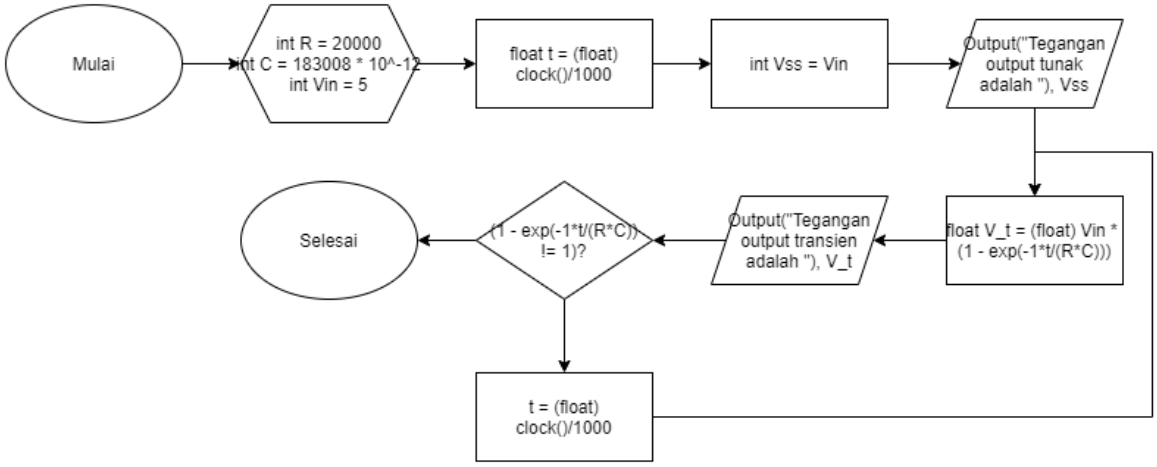
Thus,

$$v(t) = \begin{cases} V_0, & t < 0 \\ V_s + (V_0 - V_s)e^{-t/\tau}, & t > 0 \end{cases} \tag{7.46}$$

Karena Vo = 0 V, dan Vs = 5V,

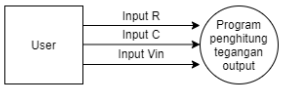
v(t) = 5 (1- e^(-t/RC))

b.

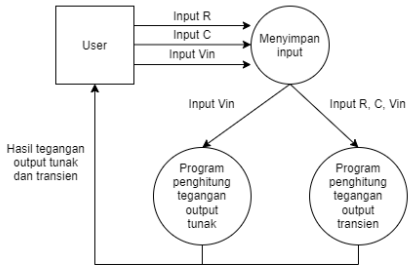


c.

Level 0



Level 1

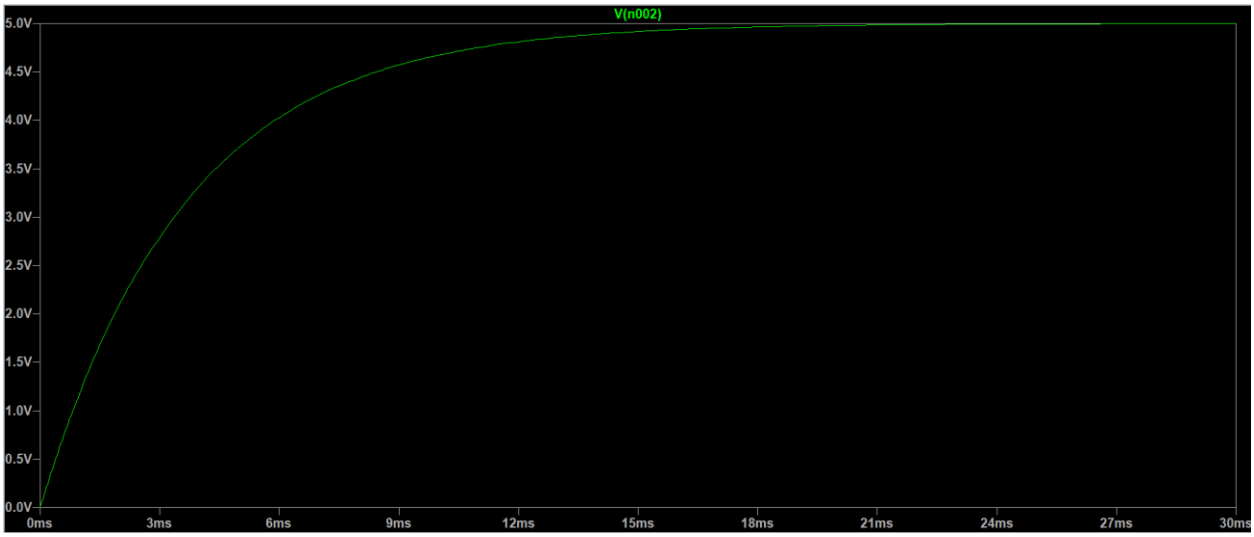


d.

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

#define R 20000
#define C 0.000000183033
#define Vin 5

int main(){
    float t = (float) clock()/1000;
    int Vss = Vin;
    printf("Tegangan output tunak adalah %d V.\n", Vss);

    while (1 - exp(-1*t/(R*C)) != 1){
        float V_t = (float) Vin * ( 1 - exp(-1*t/(R*C)));
```

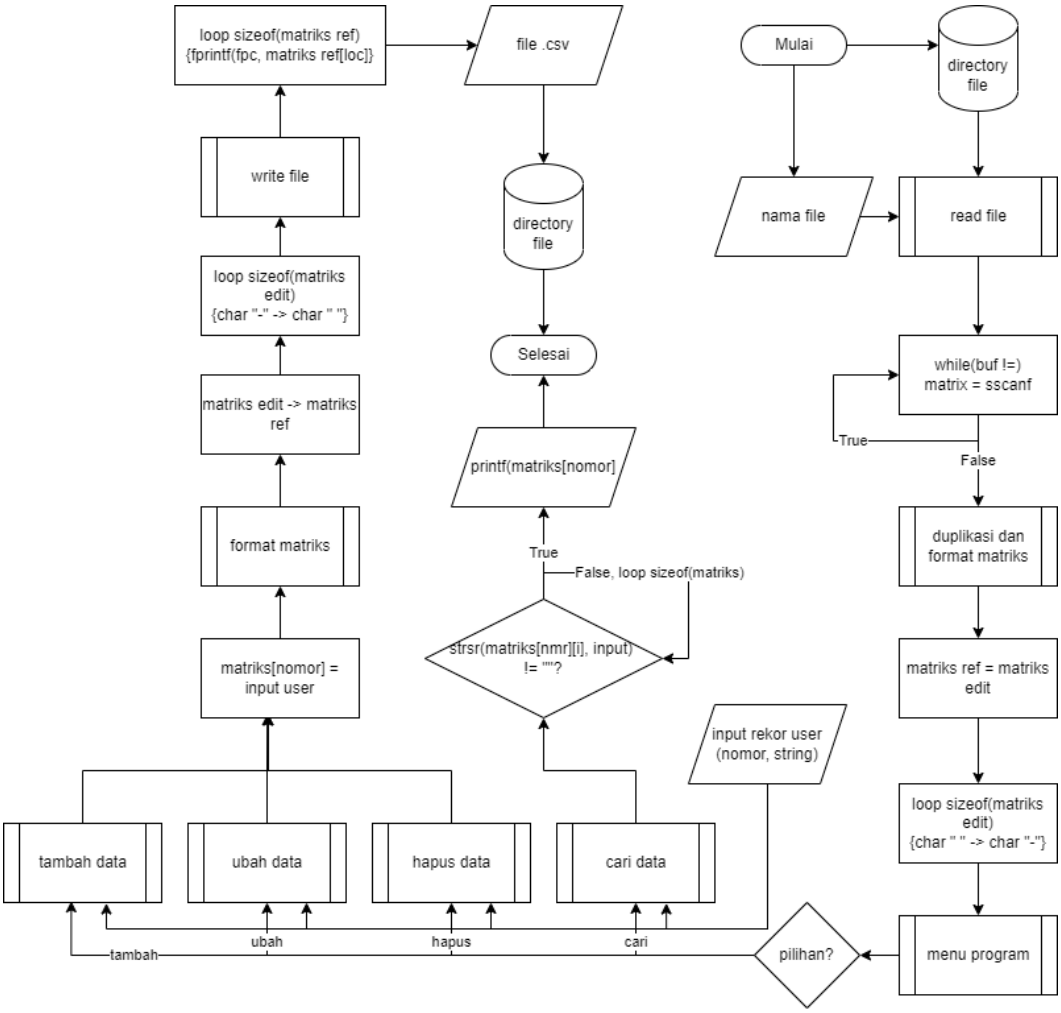
```
        printf("Tegangan output transien adalah %.2e V.\n", V_t);
        t = (float) clock()/1000;
    }
    return 0;
}
```

e.

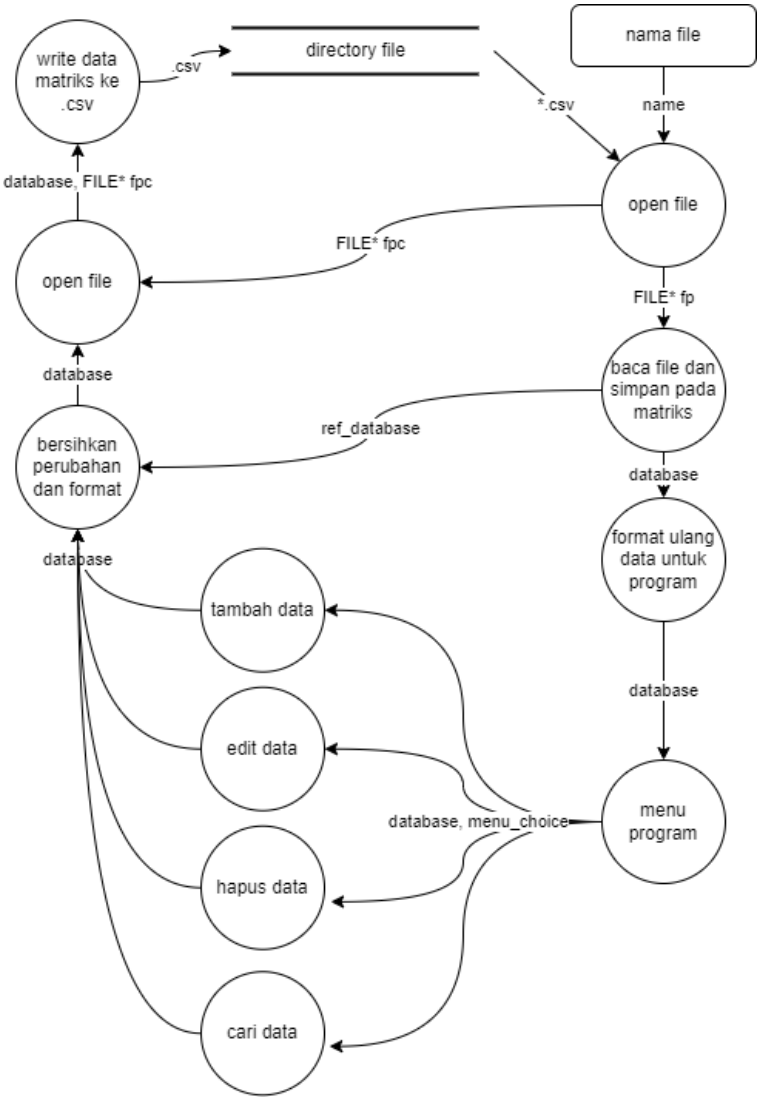


f. τ = RC = 20kOhm * 183033pF = 3.66 * 10^-3 /s, dengan program 0.003660, sama.

4a.


loop sizeof(matriks ref)
{fprintf(fpc, matriks ref[loc]}
file .csv
Mulai
directory file
write file
directory file
nama file
read file
loop sizeof(matriks edit)
{char "-" -> char " "}
Selesai
while(buf !=)
matrix = sscanf
matriks edit -> matriks ref
printf(matriks[nomor]
True
False
format matriks
True
False, loop sizeof(matriks)
duplikasi dan format matriks
matriks[nomor] = input user
strsr(matriks[nmr][i], input) != ""?
matriks ref = matriks edit
input rekor user (nomor, string)
loop sizeof(matriks edit)
{char " " -> char "-"}
tambah data
ubah data
hapus data
cari data
tambah
ubah
hapus
cari
pilihan?
menu program

b.



Skema program dapat dilihat dengan lebih jelas pada diagram alur kerja (flowchart) dan alir data (data flow diagram) berikut.

- Program pertama membuka file dengna menerima nama file serta mengakses direktori asalnya.
- Kemudian data file dibaca dan dimasukkan ke dalam dua matriks: satu sebagai referensi dan satu yang akan dieksekusi program.
- Matriks akan diolah untuk menyesuaikan format string dengan program
- Pengguna dapat memilih menu aksi yang dikerjakannya
- Pilihan pengguna dan matriks data penduduk dialihkan ke empat fungsi berbeda. Tiap fungsi masing-masing dapat mengubah, menambah, menghapus, dan mencari rekor data.
- Setelah aksi selesai, untuk fungsi yang memanipulasi data akan dibersihkan dahulu baru ditetapkan pada matriks referensi
- Program kemudian mengakses kembali pointer file asal dan menuliskan perubahannya kemudian selesai.

c.

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

typedef struct baris {
    char nik[11];
    char name[75];
    char place[30];
    char date[30];
    char age[4];
    char gender[20];
    char blood[3];
    char bond[20];
    char work[25];
} baris;


void setData(dataset) {
    char *tempArray[9];
    for (int i = 0; i < (sizeof(dataset)); i++) {
        tempArray[0] = dataset[i].nik;
        tempArray[1] = dataset[i].name;
        tempArray[2] = dataset[i].place;
        tempArray[3] = dataset[i].date;
        tempArray[4] = dataset[i].age;
        tempArray[5] = dataset[i].gender;
        tempArray[6] = dataset[i].blood;
        tempArray[8] = dataset[i].bond;
        tempArray[7] = dataset[i].work;

        for (int arrnum = 0; arrnum<9; arrnum++) {
            for (int arrloc = 0; arrloc<sizeof(tempArray[arrnum]); arrloc++)
{
                if (tempArray[0][arrloc] == " ") {
                    tempArray[0][arrloc] = "-";
                }
            }
        }

        dataset[i].nik = tempArray[0];
        dataset[i].name = tempArray[1];
        dataset[i].place = tempArray[2];
        dataset[i].date = tempArray[3];
        dataset[i].age = tempArray[4];
        dataset[i].gender = tempArray[5];
        dataset[i].blood = tempArray[6];
        dataset[i].bond = tempArray[7];
        dataset[i].work = tempArray[8];

    }
}

void menu(dataset) {
    int ask;
```

```
    printf("Silakan pilih nomor aksi Anda:\n1. Tambah rekor\n2. Ubah
rekor\n3. Hapus rekor\n4. Cari rekor\nInput: ");
    scanf("%d", &ask);
    printf("\n")

    char confirm;
    if (ask != 4 || ask != 1) {
        printf("Sudahkan cari nomor rekor aksi Anda?\ny/n ");
        scanf("%s", &confirm);
        if (confirm == "n") {
            printf("Carilah dahulu...");
            searchRec(dataset);
        }
        else if (confirm != "y") {
            printf("Konfirmasi tidak dipenuhi... Pencarian dilewatkan.");
        }
        else {
            printf("Okay");
        }
        printf("\n")
    }

    switch (ask)
    {
    case 1:
        tinkerRec(dataset, ask);
    case 2:
        tinkerRec(dataset, ask);
    case 3:
        tinkerRec(dataset, ask);
    case 4:
        searchRec(dataset);
    default:
        printf("Pilihan tidak dikenali, ulangi program.")
    }

    cleanData();

    return;
}

void searchRec(dataset)
{
    char *searchArr[9];
    printf("Cari pada rekor dengan urutan berikut, masing-masing bagian
dipisahkan dengan spasi, spasi internal dengan -\nContoh: ~ Christopher-
Columbus ~ ~\n\n");
    printf("NIK Nama-lengkap Tempat-lahir Tanggal-lahir\nInput: ");
    scanf("%s %s %s %s", &searchArr[0], &searchArr[1], &searchArr[2],
&searchArr[3]);
    printf("\nUmur Jenis-kelamin Golongan-darah Status-perkawinan
Pekerjaan\nInput: ");
    scanf("%s %s %s %s", &searchArr[4], &&searchArr[5], &searchArr[6],
&searchArr[7], &searchArr[8]);

    int count = 0;
    for (int a=0; sizeof(dataset)+sizeof(baris); a++) {
```

```
        int match = 1;

        for (int b=0; b<9; b++) {
            if (searchArr[b] != "~") {
                if (b == 0 && strstr(dataset[a].nim, searchArr[b]) != "") {
                    match = match && 1;
                }
                else if (b == 1 && strstr(dataset[a].name, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 2 && strstr(dataset[a].place, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 3 && strstr(dataset[a].date, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 4 && strstr(dataset[a].age, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 5 && strstr(dataset[a].gender, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 6 && strstr(dataset[a].blood, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 7 && strstr(dataset[a].bond, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else if (b == 8 && strstr(dataset[a].work, searchArr[b]) !=
"") {
                    match = match && 1;
                }
                else {
                    match = 0;
                }
            }
            if (match = 1) {
                count++;
                printf("\n\nTemuan %d - Nomor Rekor %d", count, a+1);
                printf("\nNIK          : %s", dataset[a].nim);
                printf("\nNama lengkap : %s", dataset[a].name);
                printf("\nTempat lahir : %s", dataset[a].place);
                printf("\nTanggal lahir: %s", dataset[a].date);
                printf("\nUsia: %s", dataset[a].age);
                printf("\nJenis kelamin: %s", dataset[a].gender);
                printf("\nGol. Darah   : %s", dataset[a].blood);
                printf("\nStat. Kawin  : %s", dataset[a].bond);
                printf("\nPekerjaan    : %s", dataset[a].work);
            }
        }
```

```
}

void addRec(dataset, int prompted)
{
    if (prompted == 1) {
        dataset = (baris *)realloc(dataset, sizeof(dataset)+sizeof(baris));

        char *searchArr[9];
        loc = sizeof(dataset)/sizeof(baris);
        printf("Masukkan rekor baru dengan urutan berikut, masing-masing
bagian dipisahkan dengan spasi, spasi internal dengan -\nContoh: 123456789012
Christopher-Columbus Bogota 29-Februari-1000\n\n");
        printf("NIK Nama-lengkap Tempat-lahir Tanggal-lahir\nInput: ");
        scanf("%s %s %s %s", &dataset.nik[loc], &dataset.name[loc],
&dataset.place[loc], &dataset.date[loc]);
        printf("\nUmur Jenis-kelamin Golongan-darah Status-perkawinan
Pekerjaan\nInput: ");
        scanf("%s %s %s %s", &dataset.age[loc], &dataset.gender[loc],
&dataset.blood[loc], &dataset.bond[loc], &dataset.work[loc]);

        }
    }
    if (prompted == 2) {
        int where;
        printf("Lokasi urutan rekor? ");
        scanf("%d", &where);

        if (where < sizeof(dataset)+sizeof(baris)) {
            printf("Ubah bagian data yang Anda inginkan, untuk bagian yang
tidak perlu letakkan $, pisahkkan antar bagian dengan spasi, spasi internal
dengan -\nContoh: $ Christopher Kolongbus $ $\n\n");
            printf("NIK Nama-lengkap Tempat-lahir Tanggal-lahir\nInput: ");
            scanf("%s %s %s %s", &dataset.nik[loc], &dataset.name[loc],
&dataset.place[loc], &dataset.date[loc]);
            printf("\nUmur Jenis-kelamin Golongan-darah Status-perkawinan
Pekerjaan\nInput: ");
            scanf("%s %s %s %s", &dataset.age[loc], &dataset.gender[loc],
&dataset.blood[loc], &dataset.bond[loc], &dataset.work[loc]);
        }
        else {
            printf("Nomor rekor tidak ada.");
        }
    }
    if (prompted == 3) {
        int where;
        printf("Lokasi urutan rekor? ");
        scanf("%d", &where);

        if (where < sizeof(dataset)+sizeof(baris)) {
            char bomb = "@";

            dataset.nik[loc], dataset.name[loc], dataset.place[loc],
dataset.date[loc], dataset.age[loc], dataset.gender[loc], dataset.blood[loc],
dataset.bond[loc], dataset.work[loc] = bomb;

        }
        else {
```

```
            printf("Nomor rekor tidak ada.");
        }
    }

}

void cleanData(baris *edited, baris *ref)
{

    char *tempArray[9];
    for (int i = 0; i < (sizeof(ref); i++)) {
        tempArray[0] = edited[i].nik;
        tempArray[1] = edited[i].name;
        tempArray[2] = edited[i].place;
        tempArray[3] = edited[i].date;
        tempArray[4] = edited[i].age;
        tempArray[5] = edited[i].gender;
        tempArray[6] = edited[i].blood;
        tempArray[8] = edited[i].bond;
        tempArray[7] = edited[i].work;

        for (int arrnum = 0; arrnum<9; arrnum++) {
            for (int arrloc = 0; arrloc<sizeof(tempArray[arrnum]); arrloc++)
{
                if (tempArray[arrnum][arrloc] == "-") {
                    tempArray[arrnum][arrloc] = " ";
                }
            }
        }

        ref[i].nik = tempArray[0];
        ref[i].name = tempArray[1];
        ref[i].place = tempArray[2];
        ref[i].date = tempArray[3];
        ref[i].age = tempArray[4];
        ref[i].gender = tempArray[5];
        ref[i].blood = tempArray[6];
        ref[i].bond = tempArray[7];
        ref[i].work = tempArray[8];

    }

    char *tempArray[9];
    for (int i = 0; i < (sizeof(ref); i++)) {
        tempArray[0] = edited[i].nik;
        tempArray[1] = edited[i].name;
        tempArray[2] = edited[i].place;
        tempArray[3] = edited[i].date;
        tempArray[4] = edited[i].age;
        tempArray[5] = edited[i].gender;
        tempArray[6] = edited[i].blood;
        tempArray[8] = edited[i].bond;
        tempArray[7] = edited[i].work;

        for (int arrnum = 0; arrnum<9; arrnum++) {

            if (tempArray[arrnum][arrloc] == "@") {
```

```
                tempArray[arrnum][arrloc] = "";
            }

        }

        ref[i].nik = tempArray[0];
        ref[i].name = tempArray[1];
        ref[i].place = tempArray[2];
        ref[i].date = tempArray[3];
        ref[i].age = tempArray[4];
        ref[i].gender = tempArray[5];
        ref[i].blood = tempArray[6];
        ref[i].bond = tempArray[7];
        ref[i].work = tempArray[8];

    }

    char *tempArray[9];
    for (int i = 0; i < (sizeof(ref); i++)) {
        tempArray[0] = edited[i].nik;
        tempArray[1] = edited[i].name;
        tempArray[2] = edited[i].place;
        tempArray[3] = edited[i].date;
        tempArray[4] = edited[i].age;
        tempArray[5] = edited[i].gender;
        tempArray[6] = edited[i].blood;
        tempArray[8] = edited[i].bond;
        tempArray[7] = edited[i].work;

        for (int arrnum = 0; arrnum<9; arrnum++) {

            if (tempArray[arrnum] == "$") {
                if (arrnum == 0) {
                    tempArray[arrnum] = ref[i].nik;
                }
                else if (arrnum == 1) {
                    tempArray[arrnum] = ref[i].name;
                }
                else if (arrnum == 2) {
                    tempArray[arrnum] = ref[i].place;
                }
                else if (arrnum == 3) {
                    tempArray[arrnum] = ref[i].date;
                }
                else if (arrnum == 4) {
                    tempArray[arrnum] = ref[i].age;
                }
                else if (arrnum == 5) {
                    tempArray[arrnum] = ref[i].gender;
                }
                else if (arrnum == 6) {
                    tempArray[arrnum] = ref[i].blood;
                }
                else if (arrnum == 7) {
                    tempArray[arrnum] = ref[i].bond;
                }
                else if (arrnum == 8) {
```

```
                    tempArray[arrnum] = ref[i].work;
                }
            }

        }

        ref[i].nik = tempArray[0];
        ref[i].name = tempArray[1];
        ref[i].place = tempArray[2];
        ref[i].date = tempArray[3];
        ref[i].age = tempArray[4];
        ref[i].gender = tempArray[5];
        ref[i].blood = tempArray[6];
        ref[i].bond = tempArray[7];
        ref[i].work = tempArray[8];

    }

}

void saveData (baris *dataset)
{

}


int main()
{
    baris *database;
    database = (baris*) malloc(sizeof(baris));

    char name[50];
    printf("Selamat datang. Silakan input nama file dan ekstensinya:\n");
    scanf("%s", &name);
    FILE* fp = fopen(name, "r");
    loadData(database, fp);

    if (fp == NULL) {
        printf("Gagal membuka file.");
        exit(EXIT_FAILURE);
    }
    char buf[200];

    int number = 0;
    while (fgets(buf, sizeof(buf), fp)) {
        database = (baris *)realloc(database,
sizeof(database)+sizeof(baris));
        sscanf(buf, "%[^;];%[^;]%[^;];%[^;]%[^;];%[^;]%[^;];%[^;];%[^;]",
&dataset[number].nik, &dataset[number].name, &dataset[number].place,
&dataset[number].date, &dataset[number].age, &dataset[number].gender,
&dataset[number].blood, &dataset[number].bond, &dataset[number].work)
        number++;
    }
    fclose(fp);

    memcpy(&database, &editbase, sizeof(database)); // matriks 1 dan matriks
2.
```

```
    setData(editbase);
    cleanData(editbase, database);

    FILE* fpc = fopen(name, "w");
    for (int loc=0; loc<sizeof(database);loc++){
        fprintf(fpc, "%d;%s;%s;%d;%d;", dataset.nik[loc], dataset.name[loc],
dataset.place[loc], dataset.date[loc], dataset.age[loc], dataset.gender[loc],
dataset.blood[loc], dataset.bond[loc], dataset.work[loc]);
        if (i!=count-1){
            fprintf(fpc, "\n");
        }
    }

    fclose(fpc);
}

    return 0;
}
```