

Praktikum Pemrograman C - Kelompok 8

SENYUMIN

PROGRAM PEMANTAUAN GIGI UNTUK
 PENGGUNA BEHEL & RETAINER



Innezahra Aurellia T.
2306168965



Mutiara Afrita Putri
2306169021



Sabina Atha Fathin
2306168984

LATAR BELAKANG

Dalam sebuah studi dengan 2644 pengguna ortodontik,

36%

Memenuhi ketaatan penuh terhadap sikat gigi dan pemakaian retainer yang tepat.

Dalam studi terhadap 60 pasien ortodonti, **aplikasi berbasis digital menunjukkan hasil terbaik** dalam menurunkan plak dan radang gusi hanya dalam waktu satu bulan.

Table 3 Comparison of mean periodontal indices for each study group at each assessment time point

Variables	Baseline				One month			
	Social media (mean \pm SD)	Assistant-based (mean \pm SD)	Software (mean \pm SD)	P value	Social media (mean \pm SD)	Assistant-based (mean \pm SD)	Software (mean \pm SD)	P value
PI	0.77 \pm 0.70	0.83 \pm 0.62	0.75 \pm 0.71	0.844	0.89 \pm 0.75 ^{ab}	1.26 \pm 0.70 ^a	0.68 \pm 0.64 ^b	0.028
GI	0.49 \pm 0.44	0.63 \pm 0.53	0.50 \pm 0.54	0.623	0.60 \pm 0.52 ^{ab}	0.92 \pm 0.67 ^a	0.46 \pm 0.52 ^b	0.047

A significant difference between groups is indicated by a different superscript letter ($p < 0.05$)

Tujuan:

- Meningkatkan kepatuhan pengguna ortodontik (behel dan retainer) dalam menyikat gigi dan memakai retainer secara rutin.
- Menjadwalkan reminder untuk kontrol ke dokter gigi sesuai periode yang ditentukan.
- Mencatat dan memantau riwayat perawatan serta keluhan secara digital dan terstruktur.

QUESTIONER

Kita melaksanakan survey singkat dengan Kautsara, mahasiswi FKG UI'22 untuk lebih memahami kebutuhan dari segi practitioner kesehatan gigi.



Kautsara FKG UI'22

Saran:

Menambahkan alur perawatan/penanganan pertama untuk keluhan pasien

1. Apa saja kendala yang sering dialami practitioner/orthodontist/dokter gigi dalam merawat pasien yang pakai behel/retainer?

- **Oral Hygiene pasien buruk** (Gigi berlubang yang dibiarkan dan tidak ditangani, karang gigi yang menumpuk, dll.)
- **Pasien tidak patuh jadwal kontrol**
- **Pasien yang berpindah-pindah dokter** (kesulitan untuk tracking rekam medis pasien)

2. Apa yang bisa dilakukan pasien untuk membantu memperlancar proses perawatan behel/retainer (dari segi practitioner)?

- **Menjaga Oral Hygiene**
- **Melaporkan segala keluhan kepada dokter gigi**
- **Retainer → menggunakan dan membersihkan secara rutin**

3. Kalau mengikuti pain scale (1-10), apakah ada angka tertentu yang menandakan pasien harus segera check-up? Atau ada parameter lain yang nentuin?

5 dari 10, serta sariawan berulang

4. Apa dampak negatif jika pasien tidak rutin kontrol, baik bagi pasien maupun practitioner?

Durasi pemakaian behel/retainer pasien menjadi lebih lama dari seharusnya, serta practitioner tidak melihat progres perkembangan.

LIMITASI

Apa saja batasan/ketidakmampuan/asumsi yang ada di program ini?

Daily Checklist

Setiap pengisian disimpan sebagai entri baru, walaupun di tanggal yang sama.

Reminder

Hanya akan tampil apabila pengguna membuka dan menjalankan program pada hari yang tepat.

Tanggal

Saat pertama kali menggunakan aplikasi, pengguna diminta memasukkan tanggal awal yang diasumsikan sebagai tanggal hari itu atau tanggal mulai penggunaan aplikasi.

Next Day

Program tidak mengikuti waktu sistem, jadi pengguna perlu menekan menu "Next Day" agar tanggal berganti.

USER

Pasien Ortodontik (Pengguna Behel Aktif)

- Individu yang sedang dalam proses perawatan ortodontik menggunakan kawat gigi (behel).
- Membutuhkan pengingat rutin untuk menyikat gigi dengan benar dan menjaga kebersihan mulut yang optimal selama perawatan.
- Memerlukan jadwal kontrol rutin ke dokter gigi untuk penyesuaian behel.
- Ingin mencatat perkembangan perawatan dan keluhan yang mungkin timbul.

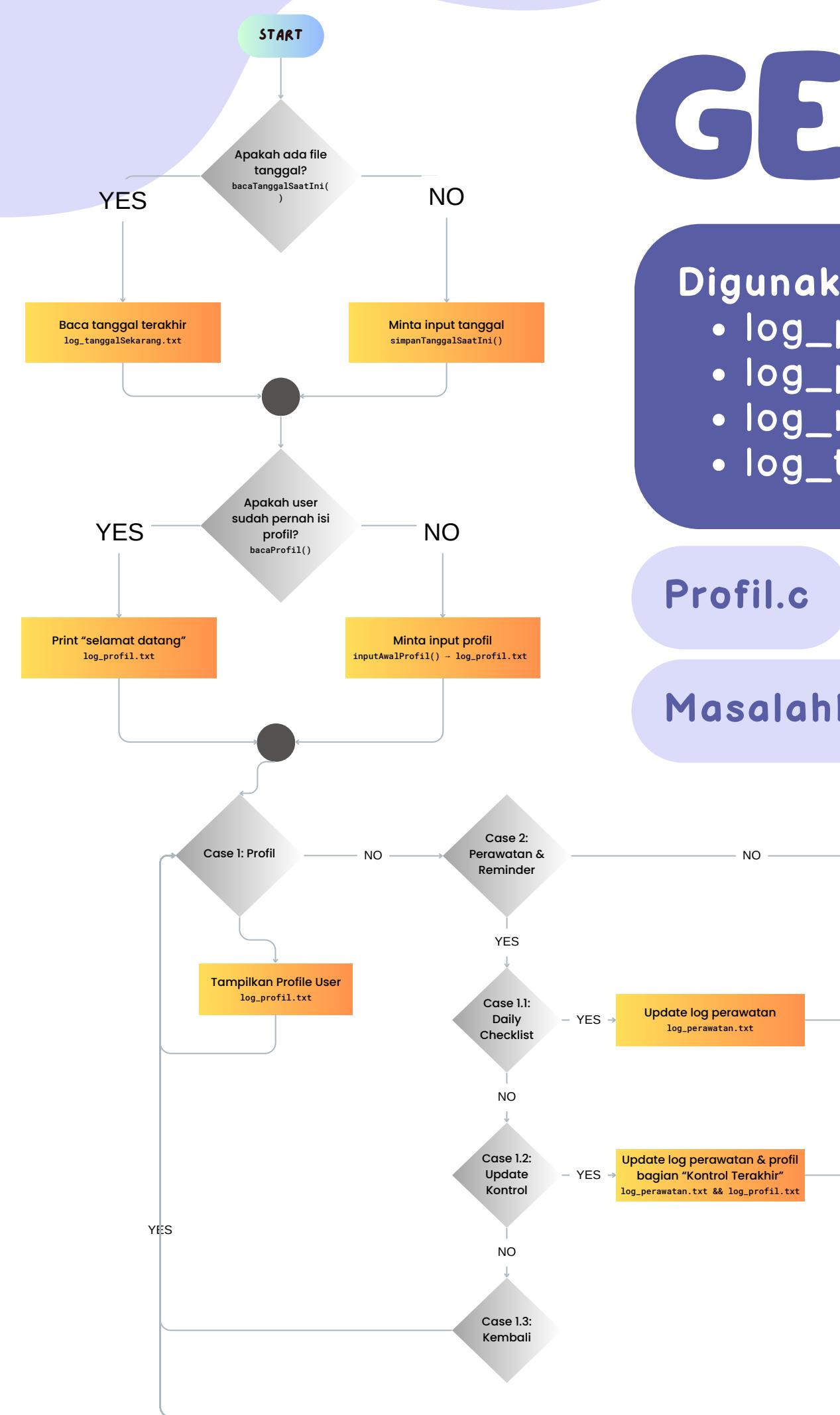
Pasien Pasca-Ortodontik (Pengguna Retainer)

- Individu yang telah menyelesaikan perawatan behel dan kini menggunakan retainer (penahan gigi) untuk menjaga posisi gigi.
- Membutuhkan pengingat untuk memakai retainer secara rutin sesuai instruksi dokter.
- Mungkin masih memerlukan kontrol berkala untuk memantau kondisi retainer atau stabilitas gigi.
- Ingin memantau riwayat penggunaan retainer dan mencatat keluhan terkait.

Orang Tua/Wali Pasien Anak/Remaja

- Bagi pasien anak-anak atau remaja, orang tua/wali akan menjadi pengguna utama yang mengelola program ini. Mereka bertanggung jawab untuk memastikan kepatuhan anak dalam menyikat gigi, memakai retainer, dan menghadiri jadwal kontrol.
- Mereka akan menggunakan fitur pencatatan dan pemantauan untuk melacak kemajuan perawatan anak mereka

GENERAL FLOWCHART



Digunakan 4 file .txt:

- log_profil.txt
- log_perawatan.txt
- log_masalah.txt
- log_tanggalSekarang.txt

libraries

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

Non-Standard Header Files

```
#include "Tanggal.h"
#include "Profile.h"
#include "PerawatanReminder.h"
#include "MasalahBantuan.h"
#include "LaporanPerawatan.h"
#include "LogKeluhan.h"
#include "ResetData.h"
#include "NextDay.h"
```

Profil.c

PerawatanReminder.c

Log Keluhan.c

NextDay.c

MasalahBantuan.c

LaporanPerawatan.c

ResetData.c

header files [file].h

enhances code functionality and readability

TANGGAL

Konversi ke Tanggal

mengonversi string tanggal ke format tm

Format Tanggal

mengonversi struktur tm kembali ke string

Simpan Tanggal Saat Ini

menyimpan tanggal saat ini ke file log tanggal sekarang.txt

Baca Tanggal Saat Ini

membaca tanggal saat ini dari file log tanggal sekarang.txt

konversiKeTanggal()

formatTanggal()

simpanTanggalSaatIni()

bacaTanggalSaatIni()

```
// Definisi variabel global tanggalSekarang
struct tm tanggalSekarang;

void konversiKeTanggal(const char *tanggalStr, struct tm *date) {
    sscanf(tanggalStr, "%d/%d/%d", &date->tm_mday, &date->tm_mon, &date->tm_year);
    date->tm_mon -= 1; // bulan mulai dari 0
    date->tm_year -= 1900; // tahun mulai dari 1900
    date->tm_hour = 0;
    date->tm_min = 0;
    date->tm_sec = 0;
}

void formatTanggal(char *buffer, struct tm date) {
    strftime(buffer, 11, "%d/%m/%Y", &date);
}

void simpanTanggalSaatIni(struct tm date) {
    FILE *fp = fopen("log_tanggalSekarang.txt", "w");
    if (fp) {
        char buf[11];
        formatTanggal(buf, date);
        fprintf(fp, "%s\n", buf);
        fclose(fp);
    }
}

int bacaTanggalSaatIni(struct tm *date) {
    FILE *fp = fopen("log_tanggalSekarang.txt", "r");
    if (!fp) return 0;
    char buf[11];
    if (fgets(buf, sizeof(buf), fp)) {
        buf[strcspn(buf, "\n")] = '\0';
        konversiKeTanggal(buf, date);
        fclose(fp);
        return 1;
    }
    fclose(fp);
    return 0;
}
```

MENU 1: PROFILE library

```
#include "Profile.h"  
#include <string.h>  
#include <stdlib.h>
```

Simpan Profile

```
void simpanProfil(struct User user) {  
    FILE *fp = fopen("log_profil.txt", "w"); // buka file, mode tulis  
  
    if (fp != NULL) {  
        // tulis data profil ke file, satu baris satu data  
        fprintf(fp, "%s\n", user.nama);  
        fprintf(fp, "%s\n", user.fase);  
        fprintf(fp, "%s\n", user.periode);  
        fprintf(fp, "%s\n", user.ortodontis);  
        fprintf(fp, "%s\n", user.tanggal_kontrol);  
        fprintf(fp, "%d\n", user.frekuensi_kontrol);  
        fclose(fp); // jangan lupa ditutup yaa  
        printf("Profil berhasil disimpan!\n");  
    } else {  
        // kalau file-nya gagal dibuka  
        printf("Gagal menyimpan profil.\n");  
    }  
}
```

Profile.h

```
#ifndef PROFILE_H  
#define PROFILE_H  
  
#include <stdio.h>  
  
// struktur buat nyimpen data profil user  
struct User {  
    char nama[50]; // nama lengkap user  
    char fase[15]; // fase (behel atau retainer)  
    char periode[30]; // periode pemakaian  
    char ortodontis[50]; // nama ortodontis, boleh kosong  
    char tanggal_kontrol[11]; // terakhir kali kontrol  
    int frekuensi_kontrol; // kontrol rutin: 30 / 45 / 60 hari  
};  
  
// Prototipe fungsi  
void simpanProfil(struct User user);  
int bacaProfil(struct User *user);  
void inputAwalProfil(struct User *user);  
  
#endif // PROFILE_H
```

Baca Profile

```
int bacaProfil(struct User *user) {  
    FILE *fp = fopen("log_profil.txt", "r"); // buka file dalam mode baca  
    if (fp == NULL) return 0; // file belum ada  
  
    // cek ukuran file, kalau 0 berarti kosong  
    fseek(fp, 0, SEEK_END);  
    long size = ftell(fp);  
    rewind(fp);  
    if (size == 0) {  
        fclose(fp);  
        return 0; // file-nya kosong  
    }  
  
    // baca data dari file, terus hapus newline di tiap input  
    fgets(user->nama, 50, fp);  
    user->nama[strcspn(user->nama, "\n")] = '\0';  
  
    fgets(user->fase, 15, fp);  
    user->fase[strcspn(user->fase, "\n")] = '\0';  
  
    fgets(user->periode, 30, fp);  
    user->periode[strcspn(user->periode, "\n")] = '\0';  
  
    fgets(user->ortodontis, 50, fp);  
    user->ortodontis[strcspn(user->ortodontis, "\n")] = '\0';  
  
    fgets(user->tanggal_kontrol, 11, fp);  
    user->tanggal_kontrol[strcspn(user->tanggal_kontrol, "\n")] = '\0';  
  
    fscanf(fp, "%d", &user->frekuensi_kontrol); // ini integer jadi pakai fscanf  
  
    fclose(fp); // done baca data  
    return 1; // profil berhasil dibaca  
}
```

MENU 1: PROFILE

```
void inputAwalProfil(struct User *user) {
    int pilihanFase;

    printf("\n===== INPUT AWAL / PROFIL =====\n");

    // input nama
    printf("Nama lengkap: ");
    fgets(user->nama, 50, stdin);
    user->nama[strcspn(user->nama, "\n")] = '\0';

    // pilih fase
    printf("Fase saat ini:\n");
    printf("[1] Pakai behel\n");
    printf("[2] Pakai retainer\n");
    printf("Pilihanmu (1/2): ");
    scanf("%d", &pilihanFase);
    getchar(); // buang newline dari buffer

    // simpan fase
    if (pilihanFase == 1) {
        strcpy(user->fase, "Behel");
    } else if (pilihanFase == 2) {
        strcpy(user->fase, "Retainer");
    } else {
        strcpy(user->fase, "Tidak Diketahui"); // fallback kalau salah input
    }
}
```

```
// input periode
printf("Periode pemakaian (riwayat): ");
fgets(user->periode, 30, stdin);
user->periode[strcspn(user->periode, "\n")] = '\0';

// input ortodontis (boleh kosong)
printf("Nama ortodontis (opsional): ");
fgets(user->ortodontis, 50, stdin);
user->ortodontis[strcspn(user->ortodontis, "\n")] = '\0';

// input tanggal kontrol terakhir
printf("Terakhir controlling (dd/mm/yyyy): ");
fgets(user->tanggal_kontrol, 11, stdin);
user->tanggal_kontrol[strcspn(user->tanggal_kontrol, "\n")] = '\0';

// input frekuensi kontrol
printf("Frekuensi kontrol rutin (30 / 45 / 60 hari): ");
scanf("%d", &user->frekuensi_kontrol);
getchar(); // bersihin newline

// simpan datanya ke file
simpanProfil(*user);
```

MENU 2: PERAWATAN & REMINDER

1. Daily Checklist

→ mencatat freq. sikat gigi & pemakaian retainer → disimpan di log_perawatan.txt

2. Update Hasil Kontrol

→ input hasil controlling → update log_profil.txt

[DAILYCHECKLIST\(\)](#)[UPDATEHASILKONTROL\(\)](#)[CEKREMINDERKONTROL\(\)](#)[SWITCH](#)[DO WHILE](#)[STRUCT](#)[POINTER](#)[POINTER](#)[ARGUMENT CALLING](#)

```
// Fungsi menu untuk memilih Daily Checklist atau Update Kontrol
void menuPerawatan(const char *fase, struct User* user) {
    int pilihan;

    do {
        printf("\nMENU PERAWATAN\n");
        printf("1. Daily Checklist\n");
        printf("2. Update Hasil Kontrol\n");
        printf("3. Kembali\n");
        printf("Pilih salah satu (1-3): ");
        scanf("%d", &pilihan);
        getchar(); // Buang newline

        switch (pilihan) {
            case 1:
                dailyChecklist(fase);
                break;
            case 2:
                updateHasilKontrol(user);
                break;
            case 3:
                printf("Kembali ke menu sebelumnya...\n");
                break;
            default:
                printf("Pilihan tidak valid. Silakan coba lagi.\n");
        }
    } while (pilihan != 3);
}
```

```
// Fungsi untuk menambahkan log ke satu file log_perawatan.txt
void tulisLog(const char *logEntry) {
    FILE *fp = fopen("log_perawatan.txt", "a"); // mode append
    if (fp != NULL) {
        fprintf(fp, "%s\n", logEntry);
        fclose(fp);
    } else {
        printf("Gagal menyimpan log ke log_perawatan.txt\n");
    }
}

// Fungsi untuk menampilkan log baru
void tampilanLogBaru(const char *judul, const char *logBaru) {
    printf("===== \n");
    printf("%s\n", judul);
    printf("%s\n", logBaru);
    printf("===== \n");
}
```

```
// Menu 1: Daily Checklist
void dailyChecklist(const char *fase) {
    char tanggal[12];
    int frekuensi;
    char log[300];

    // Ambil tanggal dari tanggalSekarang
    formatTanggal(tanggal, tanggalSekarang);
    printf("\nTanggal: %s\n", tanggal);

    printf("Frekuensi sikat gigi hari ini: ");
    scanf("%d", &frekuensi);
    getchar();

    // Tambahkan log sikat gigi
    sprintf(log, "[SIKAT GIGI] %s - %d kali", tanggal, frekuensi);
    tulisLog(log);
    tampilanLogBaru("Rekap Hari Ini - Sikat Gigi", log);

    printf("Data tersimpan.\n");

    if (frekuensi >= 2) {
        printf("Kamu sudah memenuhi target hari ini! Great Job!\n");
    } else {
        printf("Masih kurang %d kali, Jangan lupa sikat gigi!\n", 2 - frekuensi);
    }
    // Jika fase adalah Retainer, tambahkan pertanyaan
    if (strcmp(fase, "Retainer") == 0) {
        char pakaiRetainer[10];
        printf("\nSudah menggunakan retainer hari ini? (y/n): ");
        fgets(pakaiRetainer, sizeof(pakaiRetainer), stdin);
        pakaiRetainer[strcspn(pakaiRetainer, "\n")] = '\0';

        sprintf(log, "[RETAINER] %s - %s", tanggal, pakaiRetainer);
        tulisLog(log);
        tampilanLogBaru("Rekap Hari Ini - Retainer", log);

        if (strcmp(pakaiRetainer, "y") == 0 || strcmp(pakaiRetainer, "Y") == 0) {
            printf("Great Job!\n");
        } else {
            printf("Jangan lupa menggunakan retainer sebelum tidur!\n");
        }
    }
}
```

MENU 2: PERAWATAN & REMINDER

IF ELSE

ARRAY

STRUCT

POINTER

ARGUMENT CALLING

```
// Menu 2: Update hasil controlling
void updateHasilKontrol(struct User* user) {
    char tanggal[12], dokter[50], klinik[50], prosedur[100], log[300];
    char pilihan;

    // Tawarkan opsi input tanggal
    printf("\nGunakan tanggal hari ini? (y/n): ");
    scanf(" %c", &pilihan);
    while (getchar() != '\n'); // buang seluruh karakter hingga newline

    if (pilihan == 'y' || pilihan == 'Y') {
        formatTanggal(tanggal, tanggalSekarang); // dari tanggalSekarang
    } else {
        printf("Masukkan tanggal kontrol (dd/mm/yyyy): ");
        fgets(tanggal, sizeof(tanggal), stdin);
        tanggal[strcspn(tanggal, "\n")] = '\0';
    }

    printf("Nama Dokter Gigi: ");
    fgets(dokter, sizeof(dokter), stdin);
    dokter[strcspn(dokter, "\n")] = '\0';

    printf("Nama Klinik: ");
    fgets(klinik, sizeof(klinik), stdin);
    klinik[strcspn(klinik, "\n")] = '\0';

    printf("Prosedur (ngapain): ");
    fgets(prosedur, sizeof(prosedur), stdin);
    prosedur[strcspn(prosedur, "\n")] = '\0';

    sprintf(log, "[KONTROL] %s - %s/%s - %s", tanggal, dokter, klinik, prosedur);
    tulisLog(log);
    tampilanLogBaru("Rekap Hari Ini - Hasil Kontrol", log);

    // Update tanggal kontrol terakhir di profil
    if (bacaProfil(user)) {
        strcpy(user->tanggal_kontrol, tanggal);
        simpanProfil(*user);
        printf("Tanggal kontrol terakhir pada profil berhasil diperbarui.\n");
    } else {
        printf("Gagal membuka file profil.\n");
    }
}
```

STRUCT

IF ELSE

ARGUMENT CALLING

```
void cekReminderKontrol(struct User user) {
    struct tm t_kontrol;
    konversiKeTanggal(user.tanggal_kontrol, &t_kontrol);

    // Tambah frekuensi hari ke tanggal kontrol terakhir
    time_t kontrol_berikutnya = mktime(&t_kontrol) + user.frekuensi_kontrol * 86400;
    time_t sekarang = mktime(&tanggalSekarang);

    double selisih_hari = difftime(kontrol_berikutnya, sekarang) / 86400;

    if (selisih_hari <= 3 && selisih_hari >= 0) {
        if ((int)selisih_hari == 0) {
            printf("\n===== REMINDER =====\n");
            printf("Hari ini jadwal kontrol kamu! Jangan lupa datang ke klinik ya.\n");
            printf("=====\n");
        } else {
            printf("\n===== REMINDER =====\n");
            printf("Jadwal kontrol berikutnya tinggal %.0f hari lagi! Siapkan diri dan buat janji dulu\n");
            printf("=====");
        }
    }
}
```



MENU 3: MASALAH & BANTUAN

library

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

1. Pembukaan File Log

```
void masalah_dan_bantuan() {
    struct Masalah log;
    int pilihan;
    FILE *file = fopen("log_masalah.txt", "a");
    if (file == NULL) {
        printf("Gagal membuka file log_masalah.txt\n");
        return; // Keluar dari fungsi tanpa mengembalikan nilai
    }
```

2. Set tanggal + Menu keluhan

```
// Gunakan tanggal global
char tanggal[12];
formatTanggal(tanggal, tanggalSekarang);

// Menu keluhan
printf("Pilih keluhan yang kamu alami:\n");
printf("1. Sariawan\n");
printf("2. Kawat menusuk\n");
printf("3. Gusi bengkak\n");
printf("4. Nyeri rahang\n");
printf("5. Lainnya\n");
printf("Pilih nomor keluhan (1-5): ");
scanf("%d", &pilihan);
getchar(); // bersihkan newline
```

MasalahBantuan.h

```
#ifndef MASALAH_BANTUAN_H
#define MASALAH_BANTUAN_H

#include "Tanggal.h"

struct Masalah {
    char keluhan[50];
    char daerah[100];
    int tingkatSakit;
    char durasi[50];
};

void masalah_dan_bantuan();

#endif // MASALAH_BANTUAN_H
```

3. Input Tambahan

```
printf("\nDaerah yang sakit? ");
fgets(log.daerah, sizeof(log.daerah), stdin);
strtok(log.daerah, "\n");

printf("Tingkat sakit (1-10)? ");
scanf("%d", &log.tingkatSakit);
getchar();

printf("Durasi (udah berapa lama)? ");
fgets(log.durasi, sizeof(log.durasi), stdin);
strtok(log.durasi, "\n");

int parah = log.tingkatSakit >= 5;
```

4. Tampilkan + Simpan ringkasan

```
// Tampilkan ringkasan
printf("=====\\n");
printf("Ringkasan keluhan kamu:\n");
printf("Tanggal : %s\\n", tanggal);
printf("Keluhan : %s\\n", log.keluhan);
printf("Daerah : %s\\n", log.daerah);
printf("Tingkat Sakit : %d\\n", log.tingkatSakit);
printf("Durasi : %s\\n", log.durasi);
printf("=====\\n");

// Simpan ke file
fprintf(file, "\\n===== RINGKASAN MASALAH =====\\n");
fprintf(file, "Tanggal : %s\\n", tanggal);
fprintf(file, "Keluhan : %s\\n", log.keluhan);
fprintf(file, "Daerah : %s\\n", log.daerah);
fprintf(file, "Tingkat Sakit : %d\\n", log.tingkatSakit);
fprintf(file, "Durasi : %s\\n", log.durasi);
```

MENU 3: MASALAH & BANTUAN

Tips penanganan

```
// Tips penanganan
if (parah) {
    printf("Tingkat sakit tinggi! Disarankan segera ke ortodontis.\n");
    printf("\nKamu bisa mengirimkan ini ke ortho:\n");
    printf("\"Halo, saya punya keluhan %s di daerah bagian %s dengan tingkat sakit %d dan udah berlangsung selama %s\"\n",
        log.keluhan, log.daerah, log.tingkatSakit, log.durasi);
} else {
    printf("Tips penanganan awal:\n");
    switch (pilihan) {
        case 1:
            printf("- Berkumur dengan air garam hangat: Campurkan 1 sendok teh garam ke dalam segelas air hangat dan berkumur selama 30 detik, 2-3 kali sehari\n");
            printf("- Gunakan orthodontic wax: Oleskan wax pada bagian behel yang tajam untuk mengurangi gesekan dengan jaringan mulut.\n");
            printf("- Konsumsi makanan lunak: Hindari makanan keras, pedas, atau asam yang dapat memperparah sariawan\n");
            printf("- Jaga kebersihan mulut: Sikat gigi secara teratur dan gunakan benang gigi untuk mencegah infeksi\n");
            printf("- Perbanyak minum air putih: Menjaga kelembapan mulut dapat membantu proses penyembuhan.\n");
            break;
        case 2:
            printf("- Gunakan orthodontic wax: Tutup ujung kawat yang menusuk dengan wax untuk mengurangi iritasi.\n");
            printf("- Potong kawat yang berlebih: Jika memungkinkan, potong ujung kawat yang menusuk dengan gunting kuku yang bersih.\n");
            printf("- Kompres dengan es batu: Tempelkan es batu yang dibungkus kain pada area yang sakit untuk mengurangi pembengkakan.\n");
            printf("- Konsultasikan dengan ortodontis: Segera hubungi dokter gigi untuk penanganan lebih lanjut.\n");
            break;
        case 3:
            printf("- Kompres dingin: Tempelkan es batu yang dibungkus kain pada area yang Bengkak untuk mengurangi pembengkakan\n");
            printf("- Berkumur dengan air garam: Campurkan 1 sendok teh garam ke dalam segelas air hangat dan berkumur selama 30 detik.\n");
            printf("- Gunakan obat kumur antiseptik. Gunakan obat kumur antiseptik: Pilih obat kumur yang mengandung bahan antiseptik untuk membantu mengurangi peradangan.\n");
            printf("- Hindari makanan keras: Konsumsi makanan lunak untuk mengurangi tekanan pada gusi.\n");
            printf("- Jaga kebersihan mulut: Sikat gigi dengan sikat gigi khusus behel dan gunakan benang gigi secara teratur\n");
            break;
        case 4:
            printf("- Kompres hangat atau dingin: Gunakan kompres hangat atau dingin pada area rahang yang nyeri untuk meredakan ketegangan otot.\n");
            printf("- Konsumsi makanan lunak: Hindari makanan keras atau kenyal yang dapat memperparah nyeri.\n");
            printf("- Lakukan peregangan rahang: Buka dan tutup mulut perlahan untuk membantu meredakan ketegangan.\n");
            printf("- Minum obat pereda nyeri: Jika diperlukan, konsumsi obat pereda nyeri sesuai anjuran dokter.\n");
            printf("- Konsultasikan dengan ortodontis: Jika nyeri berlanjut, segera hubungi dokter gigi untuk evaluasi lebih lanjut.\n");
            break;
        case 5:
            printf("- Jaga kebersihan mulut.\n");
            printf("- Hindari makanan keras dan lengket.\n");
            printf("- Minum cukup air putih.\n");
            break;
    }
    fprintf(file, "Penanganan Awal : Lihat tips sesuai keluhan.\n");
}
fclose(file);
```

MENU 4: LAPORAN PERAWATAN

tampilkanRiwayatLengkap()

1. Profil Pengguna
→ membaca dan menampilkan informasi dasar pengguna
2. Log Perawatan
→ membaca log perawatan.txt → menampilkan riwayat kontrol
3. Log Sikat Gigi
→ membaca log perawatan.txt → menampilkan tanggal dan frekuensi sikat gigi
4. Log Pemakaian Retainer
→ membaca log perawatan.txt → menampilkan tanggal dan status

```
void tampilkanRiwayatLengkap() {
    struct User user;
    if (!bacaProfil(&user)) {
        printf("Gagal membaca profil pengguna.\n");
        return;
    }

    printf("RIWAYAT LENGKAP\n");
    printf("Nama: %s\n", user.nama);
    printf("Fase: %s\n", user.fase);
    printf("Periode Pemakaian: %s\n", user.periode);
    printf("Terakhir Controlling: %s\n", user.tanggal_kontrol);
    printf("=====\\n");

    FILE *fp = fopen("log_perawatan.txt", "r");
    if (fp == NULL) {
        printf("Belum ada log perawatan.\n");
        return;
    }

    char line[300];
```

STRUCT

IF ELSE

POINTER

ARRAY

WHILE LOOP

IF ELSE

POINTER

ARRAY

```
// Log Perawatan
printf("Log Perawatan\\n");
rewind(fp); // Kembali ke awal file
while (fgets(line, sizeof(line), fp)) {
    if (strcmp(line, "[KONTROL]", 9) == 0) {
        // Format: [KONTROL] tanggal - dokter/klinik - prosedur
        char tanggal[20], info[200];
        sscanf(line, "[KONTROL] %[^-] - %[^\\n]", tanggal, info);
        printf("%s - %s\\n", tanggal, info);
    }
}

// Log Sikat Gigi
printf("=====\\n");
printf("Log Sikat Gigi\\n");
rewind(fp);
while (fgets(line, sizeof(line), fp)) {
    if (strcmp(line, "[SIKAT GIGI]", 12) == 0) {
        char tanggal[20], frekuensi[20];
        sscanf(line, "[SIKAT GIGI] %[^-] - %[^\\n]", tanggal, frekuensi);
        printf("%s - %s\\n", tanggal, frekuensi);
    }
}

// Log Retainer
if (strcmp(user.fase, "Retainer") == 0) {
    printf("=====\\n");
    printf("Log Retainer\\n");
    rewind(fp);
    while (fgets(line, sizeof(line), fp)) {
        if (strcmp(line, "[RETAINER]", 10) == 0) {
            char tanggal[20], status[10];
            sscanf(line, "[RETAINER] %[^-] - %[^\\n]", tanggal, status);
            printf("%s - %s\\n", tanggal, status);
        }
    }
}

printf("=====\\n");
fclose(fp);
```

MENU 5: LOG KELUHAN

Menampilkan isi log_masalah.txt

```
void tampilanLogKeluhan() {
    FILE *file = fopen("log_masalah.txt", "r");
    if (file == NULL) {
        printf("Gagal membuka file log_masalah.txt\n");
        return;
    }

    char baris[200];
    char tanggal[20], keluhan[50], daerah[100], durasi[50];
    int tingkatSakit;
    int status = 0;

    printf("=====\\n");
}
```

POINTER

IF ELSE

ARRAY

DO WHILE

```
while (fgets(baris, sizeof(baris), file)) {
    if (strstr(baris, "===== RINGKASAN MASALAH =====")) {
        status = 1;
        tanggal[0] = '\0'; keluhan[0] = '\0'; daerah[0] = '\0'; durasi[0] = '\0';
        tingkatSakit = 0;
    } else if (status == 1) {
        if (strstr(baris, "Tanggal")) {
            sscanf(baris, "Tanggal      : %[^\\n]", tanggal);
        } else if (strstr(baris, "Keluahan")) {
            sscanf(baris, "Keluahan     : %[^\\n]", keluhan);
        } else if (strstr(baris, "Daerah")) {
            sscanf(baris, "Daerah      : %[^\\n]", daerah);
        } else if (strstr(baris, "Tingkat Sakit")) {
            sscanf(baris, "Tingkat Sakit : %d", &tingkatSakit);
        } else if (strstr(baris, "Durasi")) {
            sscanf(baris, "Durasi       : %[^\\n]", durasi);
        }
    }

    // Jika sudah lengkap
    if (tanggal[0] && keluhan[0] && daerah[0] && durasi[0] && tingkatSakit > 0) {
        printf("%s - %s - %s - %d/10 - %s\\n", tanggal, keluhan, daerah, tingkatSakit, durasi);
        status = 0; // Reset status
    }
}

printf("=====\\n");
fclose(file);
}
```

MENU 6: RESET DATA library

```
#include "ResetData.h"  
~~~~~  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>
```

Declare function utama

```
void resetData();  
void editProfil(struct User *user);
```

fungsi untuk reset data

```
void resetData() {  
    char konfirmasi;  
    printf("Apakah kamu yakin ingin menghapus semua data? (y/n): ");  
    scanf(" %c", &konfirmasi);  
    getchar(); // buang newline yang tersisa biar gak ganggu input selanjutnya  
  
    if (konfirmasi == 'y' || konfirmasi == 'Y') {  
        // Buka file buat dihapus isinya (mode "w" buat kosongin file)  
        FILE *f1 = fopen("log_masalah.txt", "w");  
        FILE *f2 = fopen("log_perawatan.txt", "w");  
        FILE *f3 = fopen("log_profil.txt", "w");  
        FILE *f4 = fopen("log_tanggalSekarang.txt", "w");  
        if (f1 && f2 && f3 && f4) {  
            // Kalau file terbuka dengan baik, langsung ditutup aja  
            fclose(f1);  
            fclose(f2);  
            fclose(f3);  
            fclose(f4);  
            printf("Semua data berhasil direset. Kamu bisa mulai mengisi ulang profilmu dari awal.\n");  
            exit(0); // langsung keluar dari program  
        } else {  
            // Kalau ada yang error waktu buka file  
            printf("Gagal menghapus data. Pastikan file dapat diakses.\n");  
        }  
    } else {  
        // Kalau user batal (n atau lainnya)  
        printf("Reset dibatalkan. Tidak ada data yang dihapus.\n");  
    }  
}
```

ResetData.h

```
#ifndef RESET_DATA_H  
#define RESET_DATA_H  
  
#include "Profile.h"  
  
void menuReset(struct User *user);  
  
#endif // RESET_DATA_H
```

Fungsi utama menu reset / pengaturan

```
void menuReset(struct User *user) {  
    int pilih;  
    do {  
        printf("\nMENU RESET / PENGATURAN\n");  
        printf("1. Reset Semua Data\n");  
        printf("2. Edit Profil\n");  
        printf("3. Kembali ke Menu Utama\n");  
        printf("Pilih menu (1-3): ");  
        scanf("%d", &pilih);  
        getchar();  
  
        switch(pilih) {  
            case 1:  
                resetData(); // reset data dengan konfirmasi  
                break;  
            case 2:  
                editProfil(user); // masuk ke menu edit profil  
                break;  
            case 3:  
                printf("Kembali ke menu utama...\n");  
                break;  
            default:  
                printf("Pilihan tidak valid, coba lagi.\n");  
                break;  
        }  
    } while(pilih != 3);  
}
```

MENU 6: RESET DATA

Menu edit profile

```
void editProfil(struct User *user) {
    int pilih;
    do {
        // Tampilkan menu edit profil
        printf("\n===== EDIT PROFIL =====\n");
        printf("1. Ganti Nama\n");
        printf("2. Ganti Fase (1 = Behel, 2 = Retainer)\n");
        printf("3. Ganti Periode Pemakaian\n");
        printf("4. Ganti Nama Ortodontis\n");
        printf("5. Ganti Tanggal Terakhir Kontrol\n");
        printf("6. Ganti Frekuensi Kontrol Rutin (30/45/60 hari)\n");
        printf("7. Kembali\n");
        printf("Pilih (1-7): ");
        scanf("%d", &pilih);
        getchar(); // buang newline
    }
```

```
switch(pilih) {
    case 1:
        // Ganti nama
        printf("Masukkan nama baru: ");
        fgets(user->nama, sizeof(user->nama), stdin);
        // Buang newline yang biasanya masuk karena fgets
        user->nama[strcspn(user->nama, "\n")] = '\0';
        printf("Nama berhasil diperbarui!\n");
        break;

    case 2:
        // Ganti fase behel atau retainer
        printf("Masukkan fase saat ini (1 = Behel, 2 = Retainer): ");
        int fase;
        scanf("%d", &fase);
        getchar();
        if (fase == 1) strcpy(user->fase, "Behel");
        else if (fase == 2) strcpy(user->fase, "Retainer");
        else printf("Pilihan fase tidak valid.\n");
        printf("Fase berhasil diperbarui menjadi: %s\n", user->fase);
        break;
```

```
case 3:
    // Ganti periode pemakaian
    printf("Masukkan periode pemakaian: ");
    fgets(user->periode, sizeof(user->periode), stdin);
    user->periode[strcspn(user->periode, "\n")] = '\0';
    printf("Periode berhasil diperbarui!\n");
    break;

case 4:
    // Ganti nama ortodontis
    printf("Masukkan nama ortodontis: ");
    fgets(user->ortodontis, sizeof(user->ortodontis), stdin);
    user->ortodontis[strcspn(user->ortodontis, "\n")] = '\0';
    printf("Nama ortodontis berhasil diperbarui!\n");
    break;

case 5:
    // Ganti tanggal kontrol terakhir
    printf("Masukkan tanggal terakhir kontrol (dd/mm/yyyy): ");
    fgets(user->tanggal_kontrol, sizeof(user->tanggal_kontrol), stdin);
    user->tanggal_kontrol[strcspn(user->tanggal_kontrol, "\n")] = '\0';
    printf("Tanggal kontrol berhasil diperbarui!\n");
    break;

case 6:
    // Ganti frekuensi kontrol (30, 45, atau 60 hari)
    printf("Masukkan frekuensi kontrol (30/45/60): ");
    scanf("%d", &user->frekuensi_kontrol);
    getchar();
    if(user->frekuensi_kontrol == 30 || user->frekuensi_kontrol == 45 || user->frekuensi_kontrol == 60){
        printf("Frekuensi kontrol berhasil diperbarui!\n");
    } else {
        printf("Pilihan frekuensi tidak valid.\n");
    }
    break;

case 7:
    // Kembali ke menu reset
    printf("Kembali.\n");
    break;

default:
    printf("Pilihan tidak valid, coba lagi.\n");
    break;
}

// Kalau gak pilih kembali, simpan data setelah edit
if (pilih != 7) simpanProfil(*user);

while(pilih != 7);
```

MENU 7: NEXT DAY

nextDay()

Next Day

mengubah tanggal saat ini dan memajukannya
menjadi tanggal pada hari berikutnya

```
void nextDay() {
    time_t raw = mktime(&tanggalSekarang);
    raw += 86400; // tambah 1 hari
    tanggalSekarang = *localtime(&raw);
    simpanTanggalSaatIni(tanggalSekarang);
    printf("Tanggal sekarang telah diperbarui ke hari berikutnya.\n");
}
```



Praktikum Pemrograman C - Kelompok 8

THANK YOU SENYUMIN!



REFERENSI

- [1] L. H. Timm, G. Farrag, M. Baxmann, and F. Schwendicke, "Factors Influencing Patient Compliance during Clear Aligner Therapy: A Retrospective Cohort Study," *Journal of Clinical Medicine*, vol. 10, no. 14, p. 3103, Jul. 2021, doi: <https://doi.org/10.3390/jcm10143103>.
- [2] Hooman Shafaee, S. Saeedi, Erfan Bardideh, M. Ghorbani, and P. Saeedi, "A short-term evaluation of oral hygiene education methods in fixed orthodontics patients: a randomized clinical trial comparing assistant training, software, and social media," *BMC Oral Health*, vol. 24, no. 1, Oct. 2024, doi: <https://doi.org/10.1186/s12903-024-05014-x>.
- [3] "C Tutorial," [www.w3schools.com](https://www.w3schools.com/c/index.php). <https://www.w3schools.com/c/index.php>
- [4] GeeksforGeeks, "C Programming Language," GeeksforGeeks, Aug. 29, 2024. <https://www.geeksforgeeks.org/c-programming-language/>
- [5] "Learn C - Free Interactive C Tutorial," Learn-c.org, 2019. <https://www.learn-c.org/>
- [6] Stack Overflow. <https://stackoverflow.com/questions/tagged/c>