

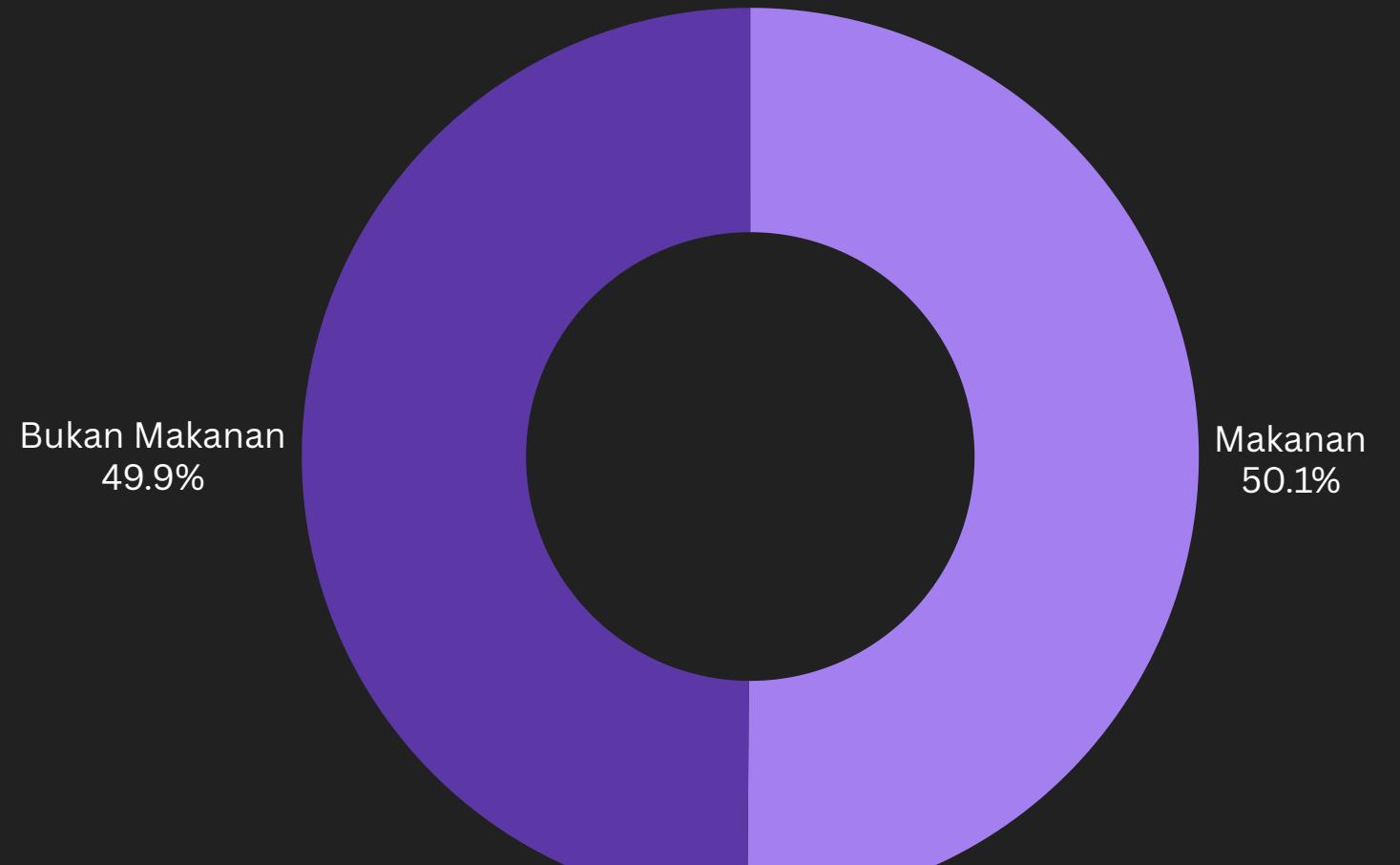
# Weekly Meal Plan

by Kelompok 1

- Argatha Advelida H. (2306168946)
- Naura Maritza D. (2306211925)



# Latar Belakang



Badan Pusat Statistik, Survei Sosial Ekonomi Nasional  
(Susenas) Maret 2024

**50,10%** pengeluaran rumah tangga digunakan untuk **makanan & minuman** dengan rata-rata Rp761.789 per kapita per bulan.



Tanpa perencanaan yang baik, alokasi pengeluaran untuk makanan menjadi tidak efisien. **Meal plan** mengatur konsumsi harian secara terstruktur dan efisien.



Ibu rumah tangga, juru masak, mahasiswa dan pekerja rantau, individu yang sedang kontrol gizi

# Tujuan

01

Membantu pengguna merencanakan menu makanan harian/mingguan secara sistematis berdasarkan resep dan bahan yang tersedia.

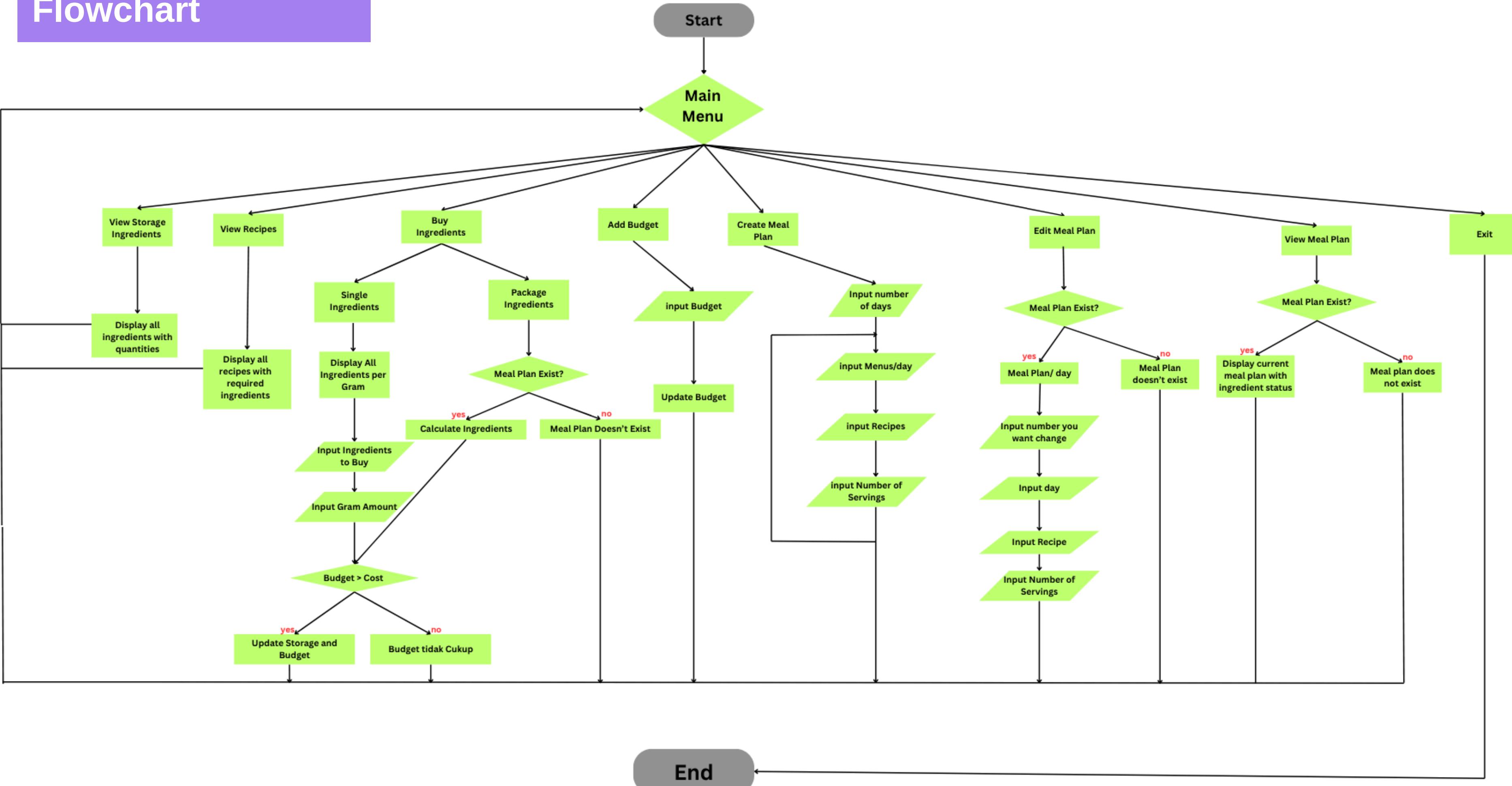
02

Menyimpan dan menampilkan bahan makanan yang dimiliki beserta harga pasarnya.

03

Membantu perencanaan anggaran belanja bahan makanan.

# Flowchart



# Kode

Kode main program untuk sistem meal plan dalam bahasa C yang mencakup menu interaktif

Library yang menyimpan seluruh fungsi yang ada pada program meal plan.

Program terus beriterasi dengan statement do while.

Menu diatur oleh statement switch case.

## Main Program

```
#include "finproFunction.h"

int main() {
    Pantry pantry;
    Recipe recipes[MAX_RECIPES];
    int choice;

    initializePantry(&pantry);
    initializeRecipes(recipes);
    printf("Selamat datang di program WEEKLY MEALPLAN\n");

    do {
        printf("\n== Menu ==\n");
        printf("1. Lihat Bahan Makanan\n");
        printf("2. Lihat Resep\n");
        printf("3. Add Budget\n");
        printf("4. Beli Bahan Makanan\n");
        printf("5. Buat Meal Plan\n");
        printf("6. Edit Meal Plan\n");
        printf("7. Lihat Meal Plan\n");
        printf("0. Exit\n");
        printf(">> ");
        scanf("%d", &choice);
        clearInputBuffer();
    }
```

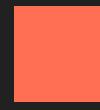
```
    switch (choice) {
        case 1:
            displayIngredients(&pantry);
            break;
        case 2:
            displayRecipes(recipes, MAX_RECIPES);
            break;
        case 3:
            addBudget(&pantry);
            break;
        case 4:
            buyIngredientsMenu(&pantry, recipes, MAX_RECIPES);
            break;
        case 5:
            createMealPlan(&pantry, recipes, MAX_RECIPES);
            break;
        case 6:
            editMealPlan(&pantry, recipes, MAX_RECIPES);
            break;
        case 7:
            displayMealPlan(&pantry, recipes, MAX_RECIPES);
            break;
        case 0:
            printf("Exiting program...\n");
            break;
        default:
            printf("Pilihan tidak valid!\n");
    }
} while (choice != 0);

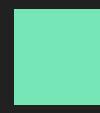
return 0;
}
```

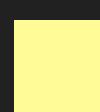
# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

 Library eksternal C yang digunakan.

 Pendefinisian nilai untuk variabel yang sering digunakan.

 Pendefinisian struktur yang digunakan pada program.

### Deklarasi & Definisi

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

#define MAX_RECIPES 7
#define MAX_NAME_LENGTH 50
#define MAX_DAYS 7
#define MAX_MENUS_PER_DAY 5
#define MAX_INGREDIENTS 20

//menyimpan nama, jumlah, dan harga bahan makanan
typedef struct {
    char name[MAX_NAME_LENGTH];
    int quantity;
    int pricePerGram;
} Ingredient;
```

```
//menyimpan bahan makanan yang diperlukan (resep) per menu
typedef struct {
    char name[MAX_NAME_LENGTH];
    Ingredient ingredients[5];
} Recipe;

//menyimpan meal plan : hari ke berapa, menu, dan porsi saji
typedef struct {
    int day;
    int recipeIndex;
    int servings;
} MealPlanEntry;

//menyimpan sistem storage
typedef struct {
    Ingredient storage[MAX_INGREDIENTS];
    int storageCount;
    int budget;
    MealPlanEntry mealPlan[MAX_DAYS * MAX_MENUS_PER_DAY];
    int mealPlanSize;
} Pantry;
```

# Kode finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

 Membuat fungsi dengan statement void.

 Menyalin isi dari source ke destination.

 Mengakses anggota dari pointer pantry.

## Clear Input Buffer

```
//menghapus sisa karakter dari input sebelumnya
void clearInputBuffer() {
    int c;
    while ((c = getchar()) != '\n' && c != EOF) {}
}
```

## Initialize Pantry

```
//inisialisasi storage awal bahan makanan
void initializePantry(Pantry *pantry) {
    pantry->storageCount = 13;
    pantry->budget = 0;

    strcpy(pantry->storage[0].name, "Daging Ayam");
    pantry->storage[0].quantity = 500;
    pantry->storage[0].pricePerGram = 300;

    strcpy(pantry->storage[1].name, "Kecap Manis");
    pantry->storage[1].quantity = 100;
    pantry->storage[1].pricePerGram = 25;

    strcpy(pantry->storage[2].name, "Bawang Merah");
    pantry->storage[2].quantity = 100;
    pantry->storage[2].pricePerGram = 50;

    strcpy(pantry->storage[3].name, "Bawang Putih");
    pantry->storage[3].quantity = 100;
    pantry->storage[3].pricePerGram = 50;

    strcpy(pantry->storage[4].name, "Cabai Merah");
    pantry->storage[4].quantity = 100;
    pantry->storage[4].pricePerGram = 44;

    strcpy(pantry->storage[5].name, "Cabai Merah Keriting");
    pantry->storage[5].quantity = 100;
    pantry->storage[5].pricePerGram = 80;
```

```
strcpy(pantry->storage[6].name, "Tomat");
pantry->storage[6].quantity = 300;
pantry->storage[6].pricePerGram = 10;

strcpy(pantry->storage[7].name, "Serai");
pantry->storage[7].quantity = 0;
pantry->storage[7].pricePerGram = 15;

strcpy(pantry->storage[8].name, "Ayam Suwir");
pantry->storage[8].quantity = 0;
pantry->storage[8].pricePerGram = 200;

strcpy(pantry->storage[9].name, "Cabai Hijau Besar");
pantry->storage[9].quantity = 0;
pantry->storage[9].pricePerGram = 150;

strcpy(pantry->storage[10].name, "Tomat Hijau");
pantry->storage[10].quantity = 0;
pantry->storage[10].pricePerGram = 25;

strcpy(pantry->storage[11].name, "Minyak Goreng");
pantry->storage[11].quantity = 100;
pantry->storage[11].pricePerGram = 10;

strcpy(pantry->storage[12].name, "Lengkuas");
pantry->storage[12].quantity = 0;
pantry->storage[12].pricePerGram = 15;

pantry->mealPlanSize = 0;
}
```

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Membuat fungsi dengan statement void.
- Program beriterasi hingga seluruh resep & bahan terdisplay.
- Menambahkan value pada budget.

### Display Recipes

```
//display menu + resep
void displayRecipes(Recipe recipes[], int recipeCount) {
    printf("\nAvailable Recipes:\n");
    for (int i = 0; i < recipeCount; i++) {
        printf("%d. %s\n", i + 1, recipes[i].name);
        printf(" Bahan:\n");
        for(int j=0;j<5;j++){
            if(strlen(recipes[i].ingredients[j].name) > 0){
                printf(" - %s: %d gr\n", recipes[i].ingredients[j].name,
                    recipes[i].ingredients[j].quantity);
            }
        }
        printf("\n");
    }
}
```

```
Available Recipes:
1. Ayam Rica-Rica
Bahan:
- Daging Ayam: 150 gr
- Cabai Merah Keriting: 20 gr
- Bawang Merah: 15 gr
- Bawang Putih: 10 gr
- Tomat: 25 gr
```

Output display recipes

### Add Budget

```
//menambahkan budget untuk beli bahan makanan
void addBudget(Pantry *pantry) {
    int additional;
    printf("Tambah budget (Rp): ");
    scanf("%d", &additional);
    clearInputBuffer();
    if (additional > 0) {
        pantry->budget += additional;
        printf("Budget berhasil ditambah, current budget: Rp %d\n",
            pantry->budget);
    } else {
        printf("Nilai tidak valid!\n");
    }
}
```

```
Tambah budget (Rp): 10000
Budget berhasil ditambah, current budget: Rp 10000
```

Output add budget

# Kode finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Membuat fungsi dengan statement void.
- Menyalin isi dari source ke destination.
- Mengakses anggota dari pointer pantry.
- Menyimpan resep dalam stuktur ingredient.

## Ingredients in storage:

1. Daging Ayam: 500 gr
2. Kecap Manis: 100 gr
3. Bawang Merah: 100 gr
4. Bawang Putih: 100 gr
5. Cabai Merah: 100 gr
6. Cabai Merah Keriting: 100 gr

Output display ingredients

## Initialize Recipe

```
//simpan resep untuk setiap menu
void initializeRecipes(Recipe recipes[]) {
    strcpy(recipes[0].name, "Ayam Rica-Rica");
    Ingredient r1[] = {{"Daging Ayam", 150,0}, {"Cabai Merah Keriting", 20,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0}, {"Tomat", 25,0}};
    memcpy(recipes[0].ingredients, r1, sizeof(r1));

    strcpy(recipes[1].name, "Ayam Bumbu Kuning");
    Ingredient r2[] = {"Daging Ayam", 150,0}, {"Kunyit", 5,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0}, {"Serai", 5,0};
    memcpy(recipes[1].ingredients, r2, sizeof(r2));

    strcpy(recipes[2].name, "Ayam Suwir Balado");
    Ingredient r3[] = {"Ayam Suwir", 150,0}, {"Cabai Merah", 25,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0}, {"Tomat", 20,0};
    memcpy(recipes[2].ingredients, r3, sizeof(r3));

    strcpy(recipes[3].name, "Ayam Kecap Pedas");
    Ingredient r4[] = {"Daging Ayam", 150,0}, {"Kecap Manis", 15,0}, {"Cabai Merah Keriting", 15,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0};
    memcpy(recipes[3].ingredients, r4, sizeof(r4));
```

```
strcpy(recipes[4].name, "Ayam Cabe Ijo");
Ingredient r5[] = {"Daging Ayam", 150,0}, {"Cabai Hijau Besar", 20,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0}, {"Tomat Hijau", 20,0};
memcpy(recipes[4].ingredients, r5, sizeof(r5));

strcpy(recipes[5].name, "Semur Ayam");
Ingredient r6[] = {"Daging Ayam", 150,0}, {"Kecap Manis", 20,0}, {"Bawang Merah", 15,0}, {"Bawang Putih", 10,0}, {"Tomat", 20,0};
memcpy(recipes[5].ingredients, r6, sizeof(r6));

strcpy(recipes[6].name, "Ayam Goreng Lengkuas");
Ingredient r7[] = {"Daging Ayam", 150,0}, {"Lengkuas", 15,0}, {"Bawang Putih", 10,0}, {"Bawang Merah", 10,0}, {"Minyak Goreng", 300,0};
memcpy(recipes[6].ingredients, r7, sizeof(r7));
}
```

## Display Ingredients

```
//menampilkan seluruh bahan makanan yang ada di storage
void displayIngredients(Pantry *pantry) {
    printf("\nIngredients in storage:\n");
    for (int i = 0; i < pantry->storageCount; i++) {
        printf("%d. %s: %d gr\n", i + 1, pantry->storage[i].name, pantry->storage[i].quantity);
    }
}
```

# Kode finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

Membuat fungsi dengan statement int yang menyimpan nilai integer.

Membandingkan ingredients yang dibutuhkan dengan yang tersedia.

Fungsi jika bahan yang dimiliki cukup.

## Check Meal Plan

```
//memeriksa adanya mealplan
int checkMealPlanCreated(Pantry *pantry) {
    return pantry->mealPlanSize > 0;
}
```

## Ingredient Index

```
//mengalokasikan index untuk setiap bahan makanan
int findIngredientIndex(Pantry *pantry, const char *name)
{
    for (int i = 0; i < pantry->storageCount; i++) {
        if (strcmp(pantry->storage[i].name, name) == 0) {
            return i;
        }
    }
    return -1;
}
```

## Update Storage

```
//menyimpan bahan makanan yang dibutuhkan untuk
//mealplan dan memeriksa kecukupannya
void updateStorageFromMealPlan(Pantry *pantry, Recipe
recipes[], int recipeCount) {
    int needed[MAX_INGREDIENTS] = {0};

    for (int i = 0; i < pantry->mealPlanSize; i++) {
        MealPlanEntry *entry = &pantry->mealPlan[i];
        Recipe *rec = &recipes[entry->recipeIndex];
        for (int ing = 0; ing < 5; ing++) {
            if (strlen(rec->ingredients[ing].name) == 0) continue;
            int idx = findIngredientIndex(pantry, rec-
>ingredients[ing].name);
            if (idx == -1) {
                strcpy(pantry->storage[pantry->storageCount].name,
rec->ingredients[ing].name);
            }
        }
    }
}
```

```
pantry->storage[pantry->storageCount].quantity = 0;
pantry->storage[pantry->storageCount].pricePerGram =
100;
idx = pantry->storageCount;
pantry->storageCount++;
}

needed[idx] += rec->ingredients[ing].quantity * entry-
>servings;
}
}

int enough = 1;
printf("\nCek kecukupan bahan untuk meal plan...\n");
for (int i = 0; i < pantry->storageCount; i++) {
    if (needed[i] > pantry->storage[i].quantity) {
        printf("- Bahan %s kurang. Butuh %d gr, tersedia %d gr\n",
pantry->storage[i].name, needed[i], pantry-
>storage[i].quantity);
        enough = 0;
    }
}

if (enough) {
    printf("Bahan cukup untuk meal plan, stok dikurangi sesuai
resep.\n");
    for (int i = 0; i < pantry->storageCount; i++) {
        pantry->storage[i].quantity -= needed[i];
    }
} else {
    printf("Bahan kurang, mohon beli bahan terlebih
dahulu.\n");
}
}
```

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Menampilkan dan memilih resep untuk membuat meal plan per hari
- Menyimpan informasi meal plan
- Validasi jumlah yang tidak valid. Tidak boleh kurang dari 1
- Mengubah isi pantry sesuai dengan meal plan yang dimasukkan

### create Meal Plan

```
void createMealPlan(Pantry *pantry, Recipe
recipes[], int recipeCount) {
    int days;
    printf("Masukkan jumlah hari untuk meal plan
(maks %d): ", MAX_DAYS);
    scanf("%d", &days);
```

```
if (days < 1 || days > MAX_DAYS) {
    printf("Jumlah hari tidak valid!\n");
    return;
}
int totalEntries = 0;
for (int d = 0; d < days; d++) {
    int menus, selRecipe, porsi;
    printf("Hari %d\n", d+1);
    printf("Berapa menu yang ingin dimasak? (maks %d): ",
MAX_MENU_PER_DAY);
    scanf("%d", &menus);

    if (menus < 1 || menus > MAX_MENU_PER_DAY) {
        printf("Jumlah menu tidak valid!\n");
        return;
    }
    for (int m = 0; m < menus; m++) {
        printf("Daftar resep:\n");
        for (int i = 0; i < recipeCount; i++) {
            printf("%d. %s\n", i + 1, recipes[i].name);
        }
        printf("Pilih resep untuk menu ke-%d: ", m + 1);
        scanf("%d", &selRecipe);

        if (selRecipe < 1 || selRecipe > recipeCount) {
            printf("Resep tidak valid!\n");
            return;
        }
        printf("Berapa porsi?: ");
        scanf("%d", &porsi);
```

```
if (porsi < 1) {
    printf("Porsi tidak valid!\n");
    return;
}
if (totalEntries >= MAX_DAYS *
MAX_MENU_PER_DAY) {
    printf("Meal plan sudah penuh!\n");
    return;
}
pantry->mealPlan[totalEntries].day = d + 1;
pantry->mealPlan[totalEntries].recipeIndex =
selRecipe - 1;
pantry->mealPlan[totalEntries].servings = porsi;
totalEntries++;
}
}
pantry->mealPlanSize = totalEntries;
updateStorageFromMealPlan(pantry, recipes,
recipeCount);
}
```

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Memeriksa apakah meal plan sudah dibuat atau belum.
- Menghitung kebutuhan bahan dari Meal Plan yang telah dibuat
- Cek kecukupan bahan di dalam penyimpanan
- Menampilkan meal plan per hari.

### display Meal Plan

```
void displayMealPlan(Pantry *pantry, Recipe
recipes[], int recipeCount) {
    if (!checkMealPlanCreated(pantry)) {
        printf("Meal plan belum dibuat.\n");
        return;
    }
```

```
// Hitung kebutuhan bahan meal plan
int needed[MAX_INGREDIENTS] = {0};
for (int i = 0; i < pantry->mealPlanSize; i++) {
    MealPlanEntry *entry = &pantry->mealPlan[i];
    Recipe *rec = &recipes[entry->recipeIndex];
    for (int ing = 0; ing < 5; ing++) {
        if (strlen(rec->ingredients[ing].name) == 0) continue;
        int idx = findIngredientIndex(pantry, rec-
>ingredients[ing].name);
        if (idx == -1) {

            // Bahan baru jika belum ada di storage
            strcpy(pantry->storage[pantry-
>storageCount].name, rec->ingredients[ing].name);
            pantry->storage[pantry->storageCount].quantity =
0;
            pantry->storage[pantry-
>storageCount].pricePerGram = 100;
            idx = pantry->storageCount;
            pantry->storageCount++;
        }
        needed[idx] += rec->ingredients[ing].quantity *
entry->servings;
    }
}
```

```
// Cek apakah bahan cukup
int enough = 1;
for (int i = 0; i < pantry->storageCount; i++) {
    if (needed[i] > pantry->storage[i].quantity) {
        enough = 0;
        break;
    }
```

```
printf("\nMeal Plan:\n");
int currentDay = -1;
for (int i = 0; i < pantry->mealPlanSize; i++) {
    if (pantry->mealPlan[i].day != currentDay) {
        currentDay = pantry->mealPlan[i].day;
        printf("Hari %d:\n", currentDay);
    }
    printf(" - %s: %d porsi\n",
        recipes[pantry->mealPlan[i].recipeIndex].name,
        pantry->mealPlan[i].servings);
}

if (!enough) {
    printf("\nCatatan: Bahan untuk meal plan ini masih
kurang, silakan beli bahan tambahan.\n");
} else {
    printf("\nBahan untuk meal plan ini cukup di
storage.\n");
}
}
```

■ Catatan ketersediaan bahan berdasarkan meal plan yang dibuat

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Menghitung total biaya berdasarkan gram x harga/gram
- Memeriksa apakah budget cukup
- Mengupdate storage dan budget setelah pembelian berhasil
- Menghitung total kebutuhan bahan untuk seluruh resep

### buySingleIngredient

```
void buySingleIngredient(Pantry *pantry, int ingredientIndex) {
    printf("Masukkan jumlah gram yang ingin dibeli untuk %s: ", pantry->storage[ingredientIndex].name);
    int qty;
    scanf("%d", &qty);
```

```
if (qty <= 0) {
    printf("Jumlah tidak valid!\n");
    return;
}
int cost = qty * pantry->storage[ingredientIndex].pricePerGram;
if (cost > pantry->budget) {
    printf("Budget tidak cukup! Total biaya: Rp %d, Budget anda: Rp %d\n", cost, pantry->budget);
    return;
}
pantry->storage[ingredientIndex].quantity += qty;
pantry->budget -= cost;
printf("Pembelian sukses! %d gr %s ditambahkan. Sisa budget: Rp %d\n",
    qty, pantry->storage[ingredientIndex].name, pantry->budget);
}
```

### buyPackagelngredient

```
void buyPackagelngredients(Pantry *pantry, Recipe recipes[], int recipeCount) {
    int needed[MAX_INGREDIENTS] = {0};
    int have[MAX_INGREDIENTS] = {0};
    int missing[MAX_INGREDIENTS] = {0};
    int totalCost = 0;

    for (int i = 0; i < pantry->storageCount; i++) {
        have[i] = pantry->storage[i].quantity;
```

```
for (int i = 0; i < pantry->mealPlanSize; i++) {
    MealPlanEntry *entry = &pantry->mealPlan[i];
    Recipe *rec = &recipes[entry->recipeIndex];
    for (int ing = 0; ing < 5; ing++) {
        if(strlen(rec->ingredients[ing].name) == 0)
            continue;
        int idx = findIngredientIndex(pantry, rec->ingredients[ing].name);
        if (idx == -1) {
            strcpy(pantry->storage[pantry->storageCount].name, rec->ingredients[ing].name);
            pantry->storage[pantry->storageCount].quantity = 0;
            pantry->storage[pantry->storageCount].pricePerGram = 100; // default harga
        }
        idx = pantry->storageCount;
        pantry->storageCount++;
    }
    needed[idx] += rec->ingredients[ing].quantity *
        entry->servings;
}
```

Menyimpan kondisi awal jumlah bahan awal untuk perbandingan

Menambahkan bahan baru jika belum ada di dalam storage

# Kode finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Menghitung bahan yang kurang, dan menentukan berapa banyak yang dibeli
- Mengupdate stok bahan dan mengurangi budget.
- Menampilkan bahan yang harus dibeli dengan harganya
- Conditional untuk membeli bahan satuan

```
for (int i = 0; i < pantry->storageCount; i++) {
    missing[i] = (needed[i] > have[i]) ? (needed[i] - have[i]) : 0;
}

printf("\nPembelian paket bahan diperlukan:\n");
for (int i = 0; i < pantry->storageCount; i++) {
    if (missing[i] > 0) {
```

```
        printf("- %s: %d gr (Rp %d)\n", pantry->storage[i].name,
               missing[i], missing[i] * pantry-
               >storage[i].pricePerGram);
        totalCost += missing[i] * pantry-
               >storage[i].pricePerGram;
    }
}

printf("Total biaya pembelian paket: Rp %d\n",
       totalCost);

if (totalCost == 0) {
    printf("Tidak ada bahan yang perlu dibeli, semua
          bahan cukup untuk meal plan.\n");
    return;
}
if (totalCost > pantry->budget) {
    printf("Budget Anda kurang untuk pembelian
          paket.\n");
    printf("Silakan tambah budget dulu.\n");
    return;
}

for (int i = 0; i < pantry->storageCount; i++) {
    if (missing[i] > 0) {
        pantry->storage[i].quantity += missing[i];
    }
}
pantry->budget -= totalCost;
printf("Pembelian paket bahan berhasil!\nSisa budget:
Rp %d\n", pantry->budget);
}
```

## buy IngredientMenu

```
void buyIngredientsMenu(Pantry *pantry, Recipe
recipes[], int recipeCount) {
    int choice;
    printf("\nMenu Beli Bahan:\n");
    printf("1. Beli bahan satuan\n");
    printf("2. Beli paket (berdasarkan meal plan
          terakhir)\n");
    printf("0. Kembali\n");
    printf("Pilih opsi: ");
    scanf("%d", &choice);

    if (choice == 1) {
        printf("\nBeli Bahan Satuan:\n");
        for (int i = 0; i < pantry->storageCount; i++) {
            printf("%d. %s (Harga Rp %d per gram)\n", i+1,
                   pantry->storage[i].name, pantry-
                   >storage[i].pricePerGram);
        }
    }

    int ingChoice;
    printf("Pilih bahan yang ingin dibeli: ");
    scanf("%d", &ingChoice);

    if (ingChoice < 1 || ingChoice > pantry->storageCount)
    {
        printf("Pilihan tidak valid!\n");
        return;
    }
```

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

Cek meal plan sudah dibuat atau belum dan pembelian paket berdasarkan meal plan

Menampilkan meal plan saat ini

Input menu yang ingin diganti, dan mengambil hari, resep dan porsi yang baru

```
}

buySingleIngredient(pantry, ingChoice - 1);
} else if (choice == 2) {
    if (!checkMealPlanCreated(pantry)) {
        printf("Meal plan belum dibuat. Silakan buat
meal plan dulu sebelum beli paket bahan.\n");
        return;
    }
}
```

```
buyPackageIngredients(pantry, recipes, recipeCount);
} else if (choice == 0) {
    return;
} else {
    printf("Pilihan tidak valid!\n");
}
}
```

### edit MealPlan

```
void editMealPlan(Pantry *pantry, Recipe recipes[], int
recipeCount) {
    if (!checkMealPlanCreated(pantry)) {
        printf("Meal plan belum dibuat. Silakan buat meal plan
terlebih dahulu.\n");
        return;
    }

    printf("Edit Meal Plan\n");
    printf("Meal plan saat ini:\n");
    for (int i = 0; i < pantry->mealPlanSize; i++) {
        printf("%d. Hari %d - %s - %d porsi\n", i + 1,
pantry->mealPlan[i].day,
recipes[pantry->mealPlan[i].recipeIndex].name,
pantry->mealPlan[i].servings);
    }

    int editIndex;
    printf("Pilih nomor menu yang ingin diubah (0 untuk
batal): ");
}
```

```
scanf("%d", &editIndex);
clearInputBuffer();
if (editIndex < 1 || editIndex > pantry->mealPlanSize) {
    printf("Pilihan invalid atau batal.\n");
    return;
}
editIndex -= 1;

int newDay, newRecipe, newServings;
printf("Pindahkan ke hari berapa (1-%d): ", MAX_DAYS);
scanf("%d", &newDay);
clearInputBuffer();
if (newDay < 1 || newDay > MAX_DAYS) {
    printf("Hari tidak valid.\n");
    return;
}
printf("Daftar resep:\n");
for (int i = 0; i < recipeCount; i++) {
    printf("%d. %s\n", i + 1, recipes[i].name);
}
printf("Pilih resep baru: ");
scanf("%d", &newRecipe);
clearInputBuffer();
if (newRecipe < 1 || newRecipe > recipeCount) {
    printf("Resep tidak valid.\n");
    return;
}
printf("Masukkan jumlah porsi baru: ");
scanf("%d", &newServings);
clearInputBuffer();
```

# Kode

## finproFunction.h

Library untuk sistem meal plan dalam bahasa C yang mencakup seluruh fungsi sistem meal plan.

- Mengembalikan bahan dari menu lama ke storage sebelum diubah
- Mengupdate storage berdasarkan meal plan baru
- Mengupdate data meal plan berdasarkan data baru

```
if (newServings < 1) {
    printf("Jumlah porsi tidak valid.\n");
    return;
}
MealPlanEntry *oldEntry = &pantry->mealPlan[editIndex];
Recipe *oldRec = &recipes[oldEntry->recipeIndex];
for (int ing = 0; ing < 5; ing++) {
```

```
    for (int ing = 0; ing < 5; ing++) {
        if (strlen(oldRec->ingredients[ing].name) == 0)
            continue;
        int idx = findIngredientIndex(pantry, oldRec->ingredients[ing].name);
        if (idx != -1) {
            pantry->storage[idx].quantity += oldRec->ingredients[ing].quantity * oldEntry->servings;
        }
    }

    pantry->mealPlan[editIndex].day = newDay;
    pantry->mealPlan[editIndex].recipeIndex = newRecipe - 1;
    pantry->mealPlan[editIndex].servings = newServings;

    updateStorageFromMealPlan(pantry, recipes,
        recipeCount);
}
```

# Limitation

---

01.

Ukuran array statis, di deklarasikan di awal sehingga tidak bisa menyesuaikan secara dinamis

02.

Hanya dapat dijalankan di *Command Line*. Sistem ini memiliki potensi untuk dibuat dalam bentuk GUI

03.

kurang Integrasi dengan Sistem Luar. Tidak dapat mengimpor langsung resep dari luar dan sinkronisasi dengan sistem luar.

04.

Meal Plan yang telah dibuat tidak dapat di save dalam bentuk file. Sehingga setelah program ditutup, meal plan akan hilang.

# Thank You.

---

