

LAPORAN PROGRESS

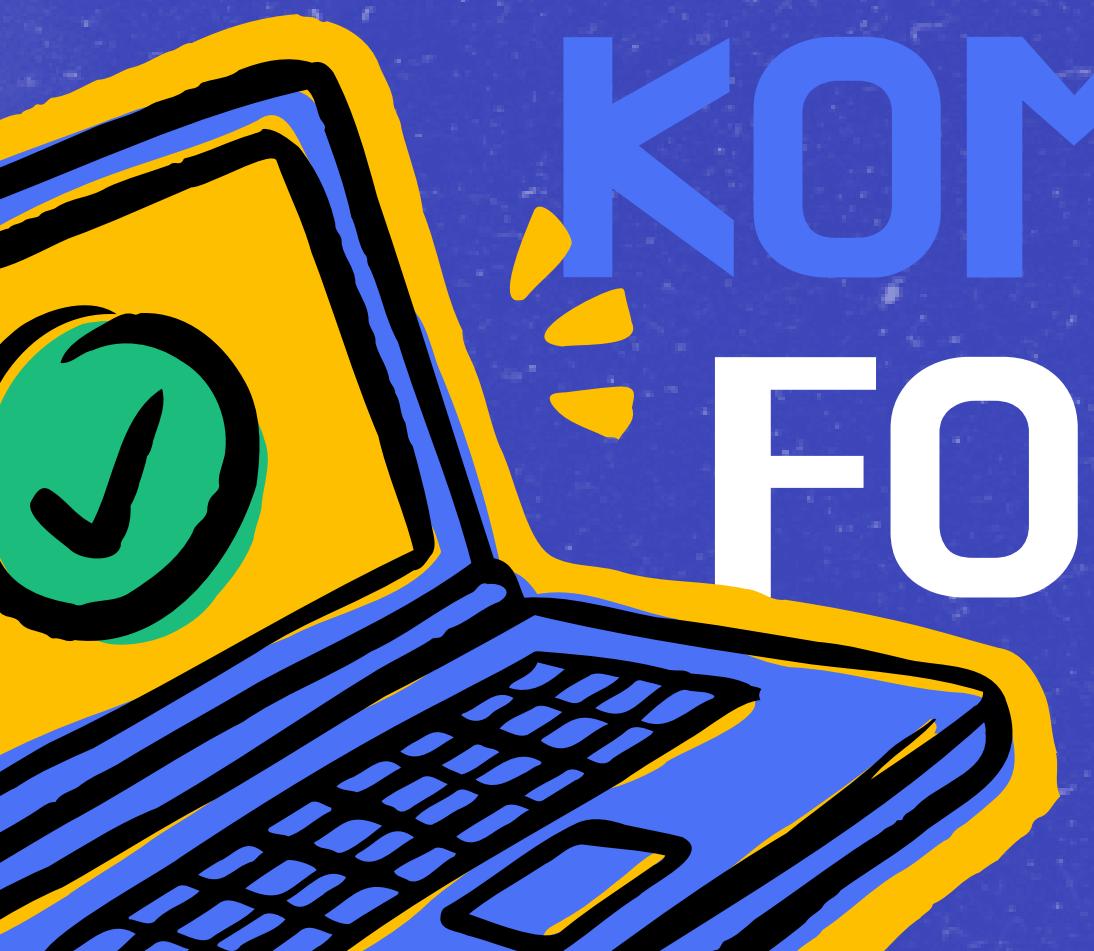
TUBES 2

MWITI102 BERPIKIR

KOMPUTASIONAL

FOOD ORDERING

KELOMPOK 2



ANGGOTA KELompok



Ishaq Irfan Fariza
19624083



Fikrifalah Muslich
19624086



Alya Nur Rahmah
19624088



Safira Berlianti
19624100



Emilio Justin
19624137

Food Ordering System

Sistem pemesanan makanan di restoran adalah teknologi yang membantu pelanggan dalam **memilih** dan **memesan** makanan secara lebih mudah dan efisien. Sistem ini memungkinkan pelanggan untuk **melihat menu**, **memilih item**, dan **memesan langsung** melalui perangkat seperti **tablet**, **aplikasi ponsel**, atau **layar digital** di restoran.



Progress keseluruhan

1. Mengefisienkan kode program dengan menggunakan subprogram
2. Merumuskan masalah pada program kami
3. Menentukan solusi yang patut dilaksanakan untuk menyelesaikan masalah
4. Membuat revisi pada flowchart sesuai dengan program yang sudah direvisi
5. Memulai progress pada laporan akhir dan ppt



Permasalahan Program

1. Kode pada program tidak efisien karena penggunaan kode yang berulang-ulang di sebagian algoritma program
2. Kode kurang mempunyai elemen yang ada pada sistem ordering pada dunia nyata

Solusi yang dilakukan

1. Menggunakan fungsi/subprogram untuk mengefisienkan kode
2. Menambahkan fitur untuk mengedit karakteristik pesanan seperti level kepedasan, ukuran minuman, dll/.
3. Menambah fitur untuk memberi tip

Revisi Aspek Berpikir Komputasional

ABSTRAKSI

Pada sistem “Food Ordering App” yang kami eksplor memiliki aspek sebagai berikut :

1. Pengisian Identitas
2. Opsi take away/dine in
3. Pemesanan jenis makan dan jumlahnya
4. Memasukkan note pesanan
5. Memberikan tip atas kepuasan layanan yang diberikan
6. Total harga dan metode pembayaran

Revisi Aspek Berpikir Komputasional

DEKOMPOSISI

1. Isi identitas

1.1 Pengisian nama

2. Pilih metode konsumsi

2.1 Take away

2.2 Dine in

2.3 Pemilihan no. meja jika dine in

3. Pilih paket makanan

3.1 Sushi

3.2 Ramen

3.3 Rice Bowl

3.4 Minuman

3.5 Paket Wibu

3.6 Appetizer

4. Penentuan jumlah pesanan

5. Menambahkan note pesanan

6. Opsi untuk memesan lagi

7. Opsi pemberian tip

8. Pilih pembayaran

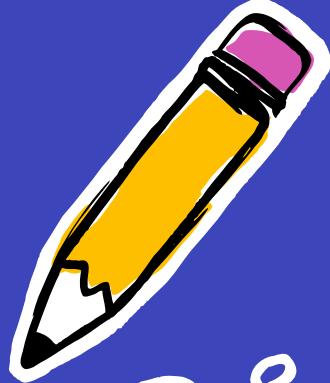
8.1 Tunai

8.2 ATM

8.3 Kredit

8.4 Q-ris

9. Cetak struk

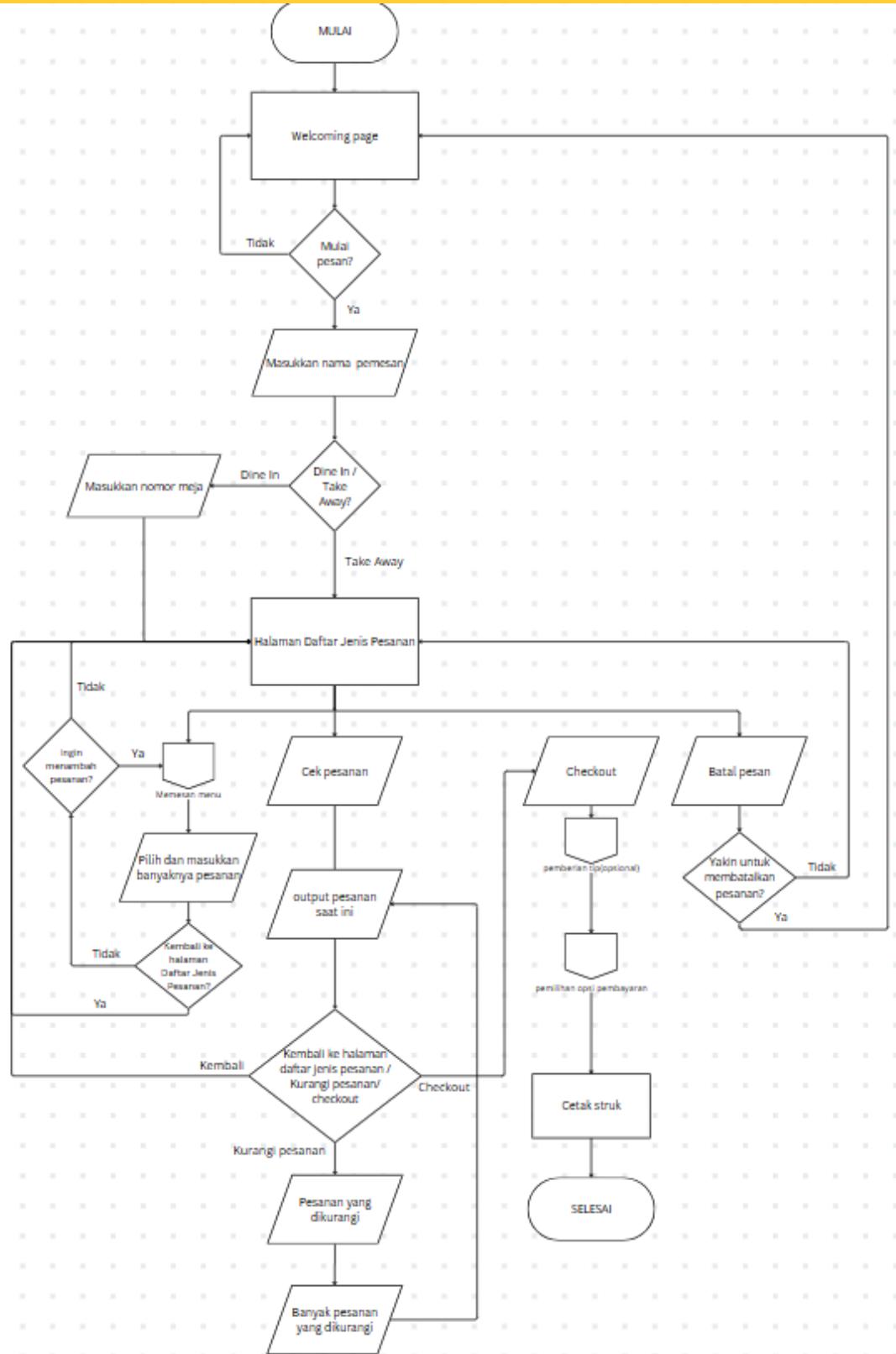


Revisi Aspek Berpikir Komputasional

PENGENALAN POLA

- Pola Pengecekan Input
- Pola untuk Menampilkan Menu Berdasarkan Kategori
- Pola Pengecekan Stok
- Pola Pemilihan Menu dan Banyaknya Berdasarkan Kategori
- Pola Pencatatan Pesanan
- Pola Pembaruan Stok
- Pola Kembali ke Halaman Utama atau Menambah Pesanan
- Pola untuk mengisi note setiap jenis pesanan

Revisi Flow Chart



Flowchart Pemberian Tip





PROGRESS PADA ALGORITMA



FITUR TIP

KODE

```
# Konfirmasi checkout
yakin_mau_checkout = True
while yakin_mau_checkout:
    yakin_checkout = input("Apakah yakin untuk checkout? (y/n): ")
    if (yakin_checkout == "y"):
        time.sleep(1)
        os.system('cls')
        print("Tip Anda akan sangat berarti untuk kami")
        tip = int(input("Rp. "))
        total_harga+= tip

# OBST PEMBAKARAN
```

TAMPILAN

```
Apakah yakin untuk checkout? (y/n): y
Tip Anda akan sangat berarti untuk kami
Rp. 100000
```

FITUR KETERANGAN

KODE

```
# Memberi keterangan untuk tambahan  
kustomisasi[indeks_pesanan] += input('Masukkan keterangan untuk tambahan untuk makanan/minuman: ')
```

TAMPILAN

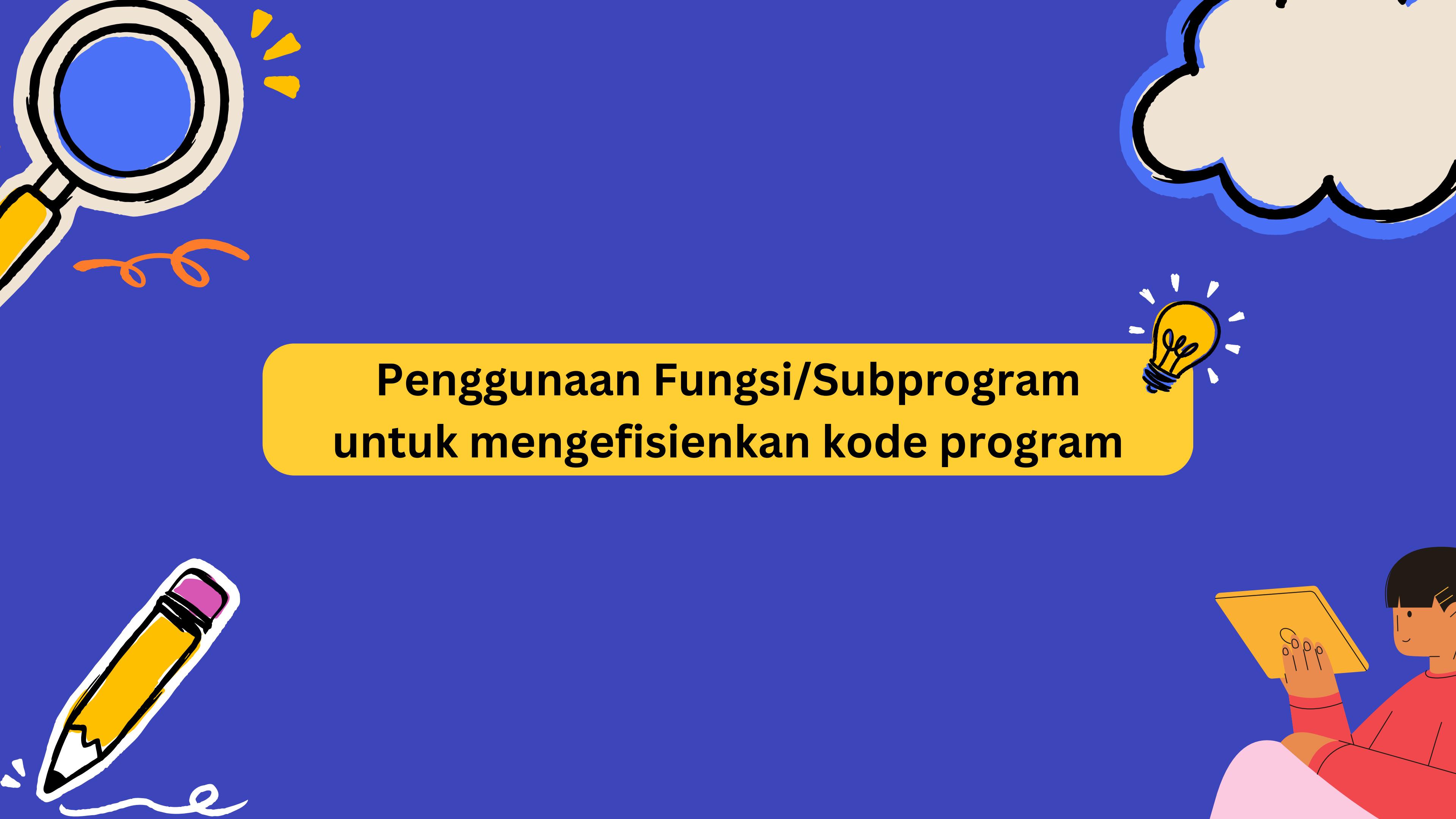
- 18. Tamago Sushi
- 19. Tamago Sushi
- 20. Salmon Cheese Roll

Pilih nomor: 2

Banyak: 2

Masukkan keterangan untuk tambahan untuk makanan/minuman:

- Rp. 27500 - Tersedia(100)-
- Rp. 11000 - Tersedia(100)-
- Rp. 55000 - Tersedia(100)-



**Penggunaan Fungsi/Subprogram
untuk mengefisienkan kode program**

SUBPROGRAM MENGECEK STOK

```
#Mengecek Stok Sushi
for i in range(sushi):
    stok_tersedia = True
    if stok_sushi[i] <= 0:
        stok_tersedia = False

    if stok_tersedia:
        stok = "Tersedia"
    else:
        stok = "Habis"
    print(f"{i+1}. {list_Sushi[i]} " * (40 - len(list_Sushi[i]) + f"{i + 1}") + f"Rp. {harga_Sushi[i]} -{stok}({stok_sushi[i]})")
```

```
#Mengecek Stok Ramen
for i in range(ramen):
    stok_tersedia = True
    if stok_ramen[i] <= 0:
        stok_tersedia = False

    if stok_tersedia:
        stok = "Tersedia"
    else:
        stok = "Habis"
    print(f"{i+1}. {list_Ramen[i]} " * (40 - len(list_Ramen[i]) + f"{i + 1}") + f"Rp. {harga_Ramen[i]} -{stok}({stok_ramen[i]})")
```

Sebelumnya algoritma program untuk mengecek masing-masing stok menu manual

B
E
F
O
R
E

SUBPROGRAM MENGECEK STOK

AFTER

```
#Fungsi yang mengecek stok sushi
def cek_stok(list_barang, stok_barang, harga_barang, jenis_menu):
    for i in range(jenis_menu):
        stok_tersedia = True
        if stok_barang[i] <= 0:
            stok_tersedia = False

        if stok_tersedia:
            stok = "Tersedia"
        else:
            stok = "Habis"
        print(f"{i+1}. {list_barang[i]} "*(40 - len(list_barang[i]) + f"{i + 1}")) + f"Rp. {harga_barang[i]} -{stok}({stok_barang[i]})-
```

Fungsi tinggal dipanggil apabila dibutuhkan dengan parameter yang sesuai

SUBPROGRAM MENGECEK PESANAN

```
#Mencatat dan mengecek pesanan yang dipilih
item = list_Sushi[pilihan - 1]
harga = harga_Sushi[pilihan - 1]

found = False
for i in range(indeks_pesanan):
    if pesanan_nama[i] == item:
        pesanan_jumlah[i] += banyak
        pesanan_total[i] += banyak * harga
        found = True
if found == False:
    pesanan_nama[indeks_pesanan] = item
    pesanan_jumlah[indeks_pesanan] = banyak
    pesanan_harga[indeks_pesanan] = harga
    pesanan_total[indeks_pesanan] = banyak * harga
    indeks_pesanan += 1

#Mengurangi stok
stok_sushi[pilihan-1] -= banyak
if (stok_sushi[pilihan-1] <= 0):
    stok_sushi[pilihan-1] = 0
```

```
#Mencatat dan mengecek pesanan yang dipilih
item = list_Ramen[pilihan - 1]
harga = harga_Ramen[pilihan - 1]

found = False
for i in range(indeks_pesanan):
    if pesanan_nama[i] == item:
        pesanan_jumlah[i] += banyak
        pesanan_total[i] += banyak * harga
        found = True
if found == False:
    pesanan_nama[indeks_pesanan] = item
    pesanan_jumlah[indeks_pesanan] = banyak
    pesanan_harga[indeks_pesanan] = harga
    pesanan_total[indeks_pesanan] = banyak * harga
    indeks_pesanan += 1
```

B
E
F
O
R
E

Sebelumnya algoritma program untuk mencatat dan mengecek pesanan menu manual

SUBPROGRAM MENGECEK PESANAN

AFTER

```
#Fungsi yang mencatat dan mengecek pesanan yang dipilih dan mengembalikan indeks_pesanan yang sudah diupdate
def mengecek_pesanan(list_barang, harga_barang, indeks_pesanan, pilihan, banyak, pesanan_nama, pesanan_jumlah, pesanan_harga, pesanan_total):

    item = list_barang[pilihan - 1]
    harga = harga_barang[pilihan - 1]

    found = False
    for i in range(indeks_pesanan):
        if pesanan_nama[i] == item:
            pesanan_jumlah[i] += banyak
            pesanan_total[i] += banyak * harga
            found = True
    if found == False:
        pesanan_nama[indeks_pesanan] = item
        pesanan_jumlah[indeks_pesanan] = banyak
        pesanan_harga[indeks_pesanan] = harga
        pesanan_total[indeks_pesanan] = banyak * harga
        indeks_pesanan += 1

    return indeks_pesanan
```

Fungsi tinggal dipanggil apabila dibutuhkan dengan parameter yang sesuai

SUBPROGRAM MENGURANGI PESANAN

```
#Mengurangi stok  
  
stok_ramen[pilihan-1] -= banyak  
  
if (stok_ramen[pilihan-1] <= 0):  
  
    stok_ramen[pilihan-1] = 0
```

```
#Mengurangi stok  
  
stok_sushi[pilihan-1] -= banyak  
  
if (stok_sushi[pilihan-1] <= 0):  
  
    stok_sushi[pilihan-1] = 0
```

Sebelumnya
algoritma
program
untuk
mengurangi
pesanan
menu
manual

BEFORE

SUBPROGRAM MENGURANGI PESANAN

AFTER

```
#Fungsi Mengurangi Stok  
  
def kurangi_stok(stok_barang, pilihan, banyak):  
  
    stok_barang[pilihan-1] -= banyak  
  
    if (stok_barang[pilihan-1] <= 0):  
        stok_barang[pilihan-1] = 0
```

Fungsi tinggal dipanggil apabila dibutuhkan dengan parameter yang sesuai

SUBPROGRAM HALAMAN UTAMA

```
#Opsi kembali ke halaman utama / tambah pesanan  
back_page = False  
  
while back_page == False:  
    go_back_page = input("Kembali ke halaman daftar jenis pesanan? (y/n): ")  
    if (go_back_page == 'n'):  
        nambah_pesanan = input("Tambah pesanan? (y/n): ")  
        print()  
        if (ambah_pesanan == 'n'):  
            pesan_lagi = False  
        elif (ambah_pesanan == 'y'):  
            os.system('cls')  
            back_page = True  
            pesan_lagi = True  
    elif (go_back_page == 'y'):  
        os.system('cls')  
        back_page = True  
        pesan_lagi = False
```

B
E
F
O
R
E

Sebelumnya algoritma program untuk kembali ke halaman utama manual

SUBPROGRAM HALAMAN UTAMA

```
#Fungsi Kembali ke halaman utama / tambah pesanan
def kembali_ke_halaman():
    back_page = False
    while back_page == False:
        go_back_page = input("Kembali ke halaman daftar jenis pesanan? (y/n): ")
        if (go_back_page == 'n'):
            nambah_pesanan = input("Tambah pesanan? (y/n): ")
            print()
            if (ambah_pesanan == 'n'):
                pesan_lagi = False
            elif (ambah_pesanan == 'y'):
                os.system('cls')
                back_page = True
                pesan_lagi = True
            elif (go_back_page == 'y'):
                os.system('cls')
                back_page = True
                pesan_lagi = False
    return pesan_lagi
```

A
F
T
E
R

Fungsi tinggal dipanggil apabila dibutuhkan dengan parameter yang sesuai

SUBPROGRAM DO-ORDER

```
#Memilih Sushi
while True:
    pilihan = input("Pilih sushi: ")
    if pilihan != "":
        pilihan = int(pilihan)
        if pilihan > 0 and pilihan < len(list_Sushi)+1:
            if stok_sushi[pilihan-1]>0:
                break
            else:
                print(f"Maaf, stok {list_Sushi[pilihan-1]} habis. Silakan pilih menu yang lain")
        else:
            print('Pastikan angka yang Anda masukkan benar.')

#Menentukan banyak sushi yang dipesan
while True:
    banyak = input("Banyak: ")
    if banyak != "":
        banyak = int(banyak)
        if banyak <= stok_sushi[pilihan-1] and banyak > 0:
            break
        else:
            print("Melebihi stok")
print()
```

B
E
F
O
R
E

Sebelumnya algoritma program untuk memesan menu manual

SUBPROGRAM HALAMAN UTAMA

```
#Fungsi gabungan untuk pemilihan menu, banyak, mengecek pesanan, dan menambah pesanan
def do_order(list_barang, stok_barang, harga_barang, indeks_pesanan, choice_jenis, total_harga):

    #Inisialisasi return value yang akan dikembalikan oleh fungsi ini, digunakan array karena akan ada 2 value yang ingin dikembalikan
    return_value = [0,0]

    #Memilih pilihan menu dan menentukan jumlahnya
    while True:
        pilihan = input("Pilih nomor: ")
        if pilihan != "":
            pilihan = int(pilihan)
            if pilihan > 0 and pilihan < len(list_barang)+1:
                if stok_sushi[pilihan-1] > 0:
                    break
                else:
                    print(f"Maaf, stok {list_barang[pilihan-1]} habis. Silakan pilih menu yang lain")
            else:
                print('Pastikan angka yang Anda masukkan benar.')

        while True:
            banyak = input("Banyak: ")
            if banyak != "":
                banyak = int(banyak)
                if banyak <= stok_barang[pilihan-1] and banyak > 0:
                    break
                else:
                    print("Melebihi stok")
        print()

    #Mengupdate indeks_pesanan jika pesanan sudah ditambahkan
    indeks_pesanan = mengecek_pesanan(list_barang, harga_barang,indeks_pesanan, pilihan, banyak, pesanan_nama, pesanan_jumlah, pesanan_harga, pesanan_total)

    #Mengurangi stok sesuai dengan jumlah pesanan yang sudah ditambahkan ke daftar pesanan
    kurangi_stok(stok_barang, pilihan, banyak)
```

Fungsi tinggal dipanggil apabila dibutuhkan dengan parameter yang sesuai

SUBPROGRAM PEMESANAN

AFTER

```
#Pemilihan Sushi
if (choice_jenis == 1):
    pesan_lagi = True

while pesan_lagi:
    print('Masukkan angka yang sesuai dengan nomor dipilih.')
    print("Pilihan Sushi:")
    print()

#Mengecek ketersediaan stok terlebih dahulu
cek_stok(list_Sushi, stok_sushi, harga_Sushi, sushi)

#Fungsi do_order() akan dijalankan dengan beberapa algoritma di dalamnya dan mengembalikan "indeks_pesanan" yang telah diupdate
indeks_dan_totalharga = do_order(list_Sushi, stok_sushi, harga_Sushi, indeks_pesanan, choice_jenis, total_harga)

indeks_pesanan = indeks_dan_totalharga[0]
total_harga = indeks_dan_totalharga[1]

#Fungsi opsi kembali ke halaman utama / tambah pesanan
pesan_lagi = kembali_ke_halaman()
```

Dengan mengimplementasikan subprogram yang telah ada, algoritma dalam setiap jenis pesanan menjadi lebih efisien dan tidak berulang



Rencana Progress Selanjutnya

- Mengusahakan algoritma program yang lebih efisien
- Membuat sistem kostumisasi yang lebih detail daripada note



**Link GitHub untuk mengakses PPT,
Laporan, dan Source Code**

<https://github.com/Valz0504/Projek-Berkom-Kelompok-2.git>



TERIMA KASIH

Terima kasih sudah
lihat presentasi kami

