

Laporan Final Berpikir Komputasional



Proyek Simulasi Food Ordering System

Kelompok 02 Kelas 29 Berpikir Komputasi :

Ishaq Irfan Farizal	19624083
Fikrifalah Muslich	19624086
Alya Nur Rahmah	19624088
Safira Berlianti	19624100
Emilio Justin	19624137

Anggota Tim dan Tugas

Nama Anggota	NIM	Tugas
Ishaq Irfan Farizal	19624083	Membuat source code, menyusun laporan akhir, dan membantu menyusun ppt progress
Fikrifalah Muslich	19624086	Berkontribusi menyusun ppt progress dan membantu membuat laporan akhir
Alya Nur Rahmah	19624088	Menyusun ppt progress dan membantu membuat laporan akhir
Safira Berlianti	19624100	Membuat flowchart, menyusun ppt progress dan membuat laporan akhir
Emilio Justin	19624137	Membuat source code, membantu menyusun laporan akhir, dan membantu menyusun ppt progress

DAFTAR ISI

DAFTAR ISI.....	iii
DAFTAR GAMBAR.....	iv
BAB I. PENDAHULUAN.....	1
A. Latar Belakang	1
B. Tujuan.....	1
BAB II. HASIL DISKUSI DAN PEMBAHASAN	2
A. Deskripsi Sistem.....	2
B. Dekomposisi	3
C. Pengenalan Pola.....	4
D. Abstraksi	5
E. Flowchart	6
F. Algoritma	7
G. Deliverable	18
BAB III. KESIMPULAN.....	19
DAFTAR PUSTAKA.....	20

DAFTAR GAMBAR

Gambar 1.1 Flowchart Program keseluruhan

Gambar 1.2 Flowchart pemesanan

Gambar 1.3 Flowchart Program memastikan pesanan

Gambar 1.4 Flowchart pembatalan pesanan

Gambar 2.1.1 Potongan kode yang menyimpan variabel data menu

Gambar 2.2.1 Tulisan selamat datang beserta tahap input informasi awal pengguna

Gambar 2.2.3 Potongan kode algoritma pengisian identitas

Gambar 2.3.1 Tampilan daftar jenis menu dan input pilihan

Gambar 2.3.2 Tampilan daftar jenis makanan dan minuman

Gambar 2.3.3 Tampilan input lanjutan setelah pesan

Gambar 2.3.4 Potongan kode algoritma tampilan list jenis makanan

Gambar 2.3.5 Potongan kode algoritma pemilihan, pengecekan stok, dan memilih makanan atau minuman (sampel yang diambil “Sushi”)

Gambar 2.3.6 Potongan kode algoritma penentuan banyak, pencatatan, dan pengecekan makanan atau minuman

Gambar 2.3.7 Potongan kode algoritma pengurangan stok dan opsi kembali ke halaman utama/tambah pesanan

Gambar 2.4.1 Tampilan jika pengguna tidak memesan

Gambar 2.4.2 Tampilan cek pesanan dan pilihan kurangi pesanan

Gambar 2.4.3 Potongan kode algoritma menampilkan pesanan saat ini dan pemberian opsi untuk pengguna

Gambar 2.4.4 Potongan kode algoritma mengurangi pesanan

Gambar 2.4.5 Potongan kode algoritma mengembalikan stok apabila ada menu yang dikurangi

Gambar 2.5.1 Tampilan Batal Pesan

Gambar 2.5.2 Potongan kode algoritma batal pesan dan pengembalian stok apabila pesanan dibatalkan

Gambar 2.6.1 Tampilan opsi pembayaran

Gambar 2.6.2 Potongan kode algoritma konfirmasi checkout dan pemilihan opsi pembayaran

Gambar 2.6.3 Potongan kode algoritma pencetakan struk pesanan (sampel yang diambil pembayaran selain Q-RIS)

Gambar 2.6.4 Potongan kode algoritma konfirmasi pembayaran

BAB I

PENDAHULUAN

A. Latar Belakang

Berpikir komputasional adalah pendekatan yang digunakan untuk memecahkan masalah kompleks dengan cara logis dan kalkulatif. Berpikir komputasional sebenarnya tidak mengarah kepada penggunaan komputer atau software, tetapi lebih terhadap pola pikir yang mementingkan pengupasan masalah secara detail dan cara sistematis penyelesaian masalah tersebut.

Berpikir komputasional terbagi menjadi 4 aspek, yakni abstraksi, dekomposisi, pengenalan pola, dan algoritma. Abstraksi berarti mengambil informasi yang krusial dan membuang informasi yang tidak penting. Dekomposisi berarti memecahkan masalah menjadi masalah-masalah kecil. Pengenalan pola berarti mengidentifikasi pola yang ada pada masalah. Algoritma berarti prosedur atau langkah langkah dalam pemecahan masalah.

Dalam era digitalisasi, berpikir komputasional melatih kita agar dapat memaksimalkan maraknya pengembangan digital untuk memecahkan masalah. Selain hanya dipelajari dan direvisi, keterampilan yang diajarkan konsep berpikir komputasional patutnya di praktikan dan dilatih. Salah satu cara yang yang bisa dilakukan adalah pengamatan dan pembuatan replikasi sistem-sistem yang ada di dunia. Hal ini membuat kita lebih mengerti perbedaan algoritma yang efisien dan efektif dibanding dengan yang tidak. Selain itu, dengan melakukan pengamatan, kita bisa melatih keterampilan kita untuk melakukan abstraksi, dekomposisi, dan pengenalan pola.

B. Tujuan

1. Melakukan abstraksi, dekomposisi, dan pengenalan pola pada sistem agar mempunyai data gambaran konkret akan sistem yang diamati
2. Membuat simulasi algoritma sistem yang diamati dengan menggunakan bahasa pemrograman python

BAB II

HASIL DISKUSI DAN PEMBAHASAN

A. Deskripsi Sistem

Food Ordering System adalah sebuah sistem pemesanan makanan yang dirancang untuk memudahkan pelanggan dalam memilih dan memesan hidangan di restoran, sekaligus mempermudah restoran dalam mencatat pesanan pelanggan dan menghindari adanya kesalahan. Sistem ini memungkinkan pelanggan untuk melihat serta memilih hidangan yang diinginkan, menambahkannya ke dalam daftar pesanan, dan memodifikasi pilihan mereka dengan cepat seperti menambah atau mengurangi jumlah pesanan sesuai keinginan. Dengan menggunakan sistem ini, pemesanan makanan pada suatu restoran menjadi lebih efektif dan efisien.

Sistem “*Food Ordering System*” yang kami rancang ini memiliki beberapa fungsionalitas untuk memudahkan pemesanan makanan pada suatu restoran. Fungsionalitas tersebut diantaranya adalah mencatat identitas pelanggan seperti nama dan nomor meja apabila memilih opsi dine in; menampilkan berbagai menu yang dapat dipilih dilengkapi dengan harga dan stok yang tersedia sehingga pelanggan dapat mengetahui menu apa saja yang bisa dipesan; menambah ataupun mengurangi jumlah pesanan sesuai dengan keinginan pelanggan; halaman check out atau pengecekan kembali menu apa saja yang telah dipesan lengkap dengan jumlah serta harga total dan juga pelanggan dapat bebas memilih membayar pesanan menggunakan metode pembayaran yang diinginkan.

Sistem yang kami rancang ini dibuat dengan menggunakan teknik berpikir komputasi. Teknik tersebut antara lain dimulai dari abstraksi sistem sebagai awal untuk menentukan komponen utama pada sistem yang kami rancang agar sesuai dengan fungsinya sebagai sistem program untuk memesan makanan pada suatu restoran; kemudian setelah ditentukan komponen utama pada sistem dilakukan dekomposisi untuk memecah komponen tersebut menjadi beberapa bagian sesuai dengan fungsionalitasnya; diterapkan pengenalan pola pada sistem yang kami rancang ini untuk melihat pola yang digunakan pada sistem serta memudahkan dalam pembuatan program; yang terakhir adalah menentukan algoritma yang digunakan dalam hal ini sistem kami ditulis menggunakan bahasa pemrograman python dengan menggunakan aspek dasar pemrograman seperti sekuens, kondisional, perulangan, dan array.

B. Dekomposisi Sistem

Dekomposisi sistem adalah cara pemecahan masalah menjadi bagian yang lebih kecil ketika menghadapi masalah yang kompleks. Salah satu cara pendekatan untuk memecahkan masalah tersebut adalah dengan memecahnya menjadi sub-masalah yang lebih kecil dan lebih mudah dipahami dan dikelola. Dengan memahami bagian-bagian kecil dari masalah, dapat lebih mudah menyusun sebuah solusi. Berikut ini dekomposisi dari sistem “*Food Ordering Application*” yang telah kami rancang :

1. 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. Pemilihan paket makanan atau minuman
 - 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. Opsi untuk memesan lagi dan menambah atau mengurangi pesanan
6. Pencatatan Pesanan yang telah dibuat
7. Metode pembayaran
 - 6.1 Tunai
 - 6.2 ATM
 - 6.3 Kredit
 - 6.4 Q-ris
8. Cetak struk dan pemrosesan pesanan

C. Pengenalan Pola Sistem

Pengenalan pola pada suatu sistem dilakukan dengan cara mencari kesamaan yang kemudian dapat digunakan untuk mengidentifikasi serta menyelesaikan sistem yang akan dirancang. Algoritma *food ordering system* yang telah kami rancang menerapkan beberapa pola berikut ini, antara lain:

1. Pola Pengecekan Input

Pola pengecekan input yaitu pola untuk memastikan apakah input dari pengguna valid. Jika input dari pengguna tidak valid, pengguna akan diminta untuk memasukkan input kembali hingga inputnya benar. Proses pemesanan hanya dapat dilanjutkan setelah pengguna memasukkan input yang benar. Pola ini digunakan untuk mencegah kesalahan input yang dapat menyebabkan error.

2. Pola untuk Menampilkan Menu Berdasarkan Kategori

Setiap kali pengguna memilih kategori menu pada halaman daftar jenis menu, program akan menampilkan menu berdasarkan kategori yang dipilih pengguna tersebut. Selain menu, program juga akan menampilkan harga dan status stok setiap menu. Pola ini ada pada algoritma setiap kategori menu.

3. Pola Pengecekan Stok

Pola pengecekan stok yaitu pola untuk mengecek apakah stok masih tersedia. Pola ini ada pada algoritma setiap kategori menu. Pola ini digunakan untuk menampilkan status stok ("Tersedia" atau "Habis") kepada pengguna dan memastikan agar pengguna tidak bisa memesan menu yang stoknya habis.

4. Pola Pemilihan Menu dan Banyaknya Berdasarkan Kategori

Setelah pengguna memilih kategori menu pada halaman daftar jenis menu, pengguna dapat memilih menu yang tersedia dan banyak menu yang dipesan berdasarkan kategori yang dipilih. Pola ini ada pada algoritma setiap kategori menu.

5. Pola Pencatatan Pesanan

Setelah pengguna memilih menu, program akan mencatat menu yang dipesan, banyaknya pesanan, dan total harga pesanan. Pola ini ada pada algoritma setiap kategori menu. Pola ini berguna untuk menyimpan riwayat pesanan yang dapat digunakan untuk cek pesanan dan mencetak struk pesanan.

6. Pola Pembaharuan Stok

Pola ini akan membuat stok berkurang setiap kali pengguna menambah pesanan. Pola ini ada pada algoritma setiap kategori menu. Selain itu, program juga akan mengembalikan stok sebanyak pesanan yang dikurangi setiap kali pengguna mengurangi pesanan. Apabila pengguna membatalkan pesanan, stok akan kembali seperti semula.

7. Pola Kembali ke Halaman Utama atau Menambah Pesanan

Setiap pengguna telah memesan, akan terdapat pilihan untuk kembali ke halaman utama atau tambah pesanan. Pola ini ada pada algoritma setiap kategori menu. Jika pengguna memilih kembali ke halaman utama, pengguna akan berpindah ke halaman daftar jenis makanan. Jika pengguna memilih tambah pesanan, pengguna akan tetap berada pada halaman menu berdasarkan kategori yang dipilih dan dapat menambah pesanan berdasarkan kategori yang dipilih. Pola ini memungkinkan pengguna untuk menambah pesanan tanpa perlu mengulangi proses dari awal.

D. Abstraksi

Abstraksi pada sistem adalah memfokuskan pada informasi penting di mana tidak semua detail dari sebuah masalah tersebut perlu kita perhatikan. Abstraksi memungkinkan kita untuk menyaring informasi yang tidak relevan dan fokus pada detail-detail yang penting saja. Beberapa komponen penting pada “*Food Ordering System*” yang telah kami rancang antara lain sebagai berikut :

1. Identitas Pemesan

Identitas pemesan hanya mencakup nama pemesan, yang berguna untuk memverifikasi pemesanan dan memastikan pesanan disiapkan untuk orang yang tepat.

2. Metode Pemesanan

Terdapat dua metode pemesanan, yaitu “Dine In” dan “Take Away”. Metode pemesanan “Dine In” melibatkan pemesanan nomor meja, sedangkan “Take Away” menandakan bahwa pesanan perlu dikemas untuk dibawa oleh pelanggan.

3. Menu

Dengan adanya menu, pengguna dapat mengetahui daftar menu beserta harga dan ketersediaan stoknya. Ada enam kategori menu pada *food ordering system* kami, yaitu sushi, ramen, rice bowl, minuman, Paket Wibu, dan appetizer.

4. Harga

Harga diperlukan untuk mengetahui biaya untuk setiap item di menu dan menghitung total harga dari menu yang dipesan.

5. Stok

Stok menyatakan jumlah ketersediaan dari setiap *item* pada menu. Stok diperlukan untuk memastikan bahwa hanya item yang tersedia saja yang dapat dipesan.

6. Pesanan

Pesanan mencakup menu yang dipesan dan banyaknya. Informasi ini berguna untuk menghitung total harga dan mencetak struk.

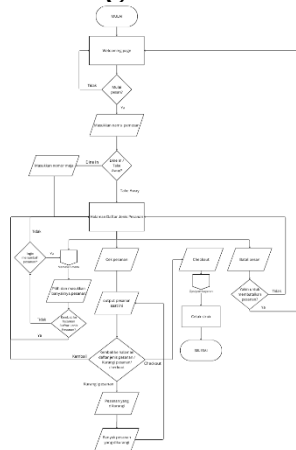
7. Total harga

Total harga diperlukan untuk menginformasikan jumlah total yang harus dibayar oleh pelanggan untuk pesanan mereka. Total harga dapat dihitung berdasarkan harga per *item* dan jumlah *item* yang dipesan oleh pelanggan.

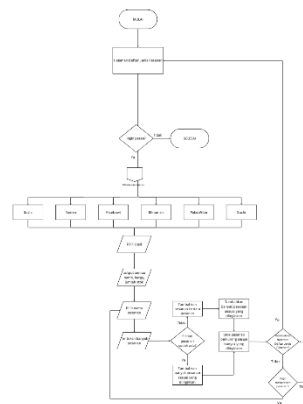
8. Pembayaran

Terdapat empat metode pembayaran pada *food ordering system* kami, yaitu tunai, ATM, kredit, dan QRIS. Pembayaran berperan sebagai tahap akhir untuk menyelesaikan transaksi.

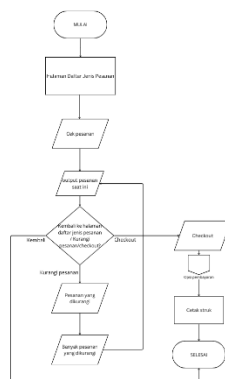
E. Flowchart Program



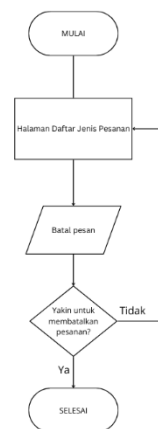
Gambar 1.1 Flowchart Program keseluruhan



Gambar 1.2 Flowchart pemesanan



Gambar 1.3 Flowchart Program memastikan pesanan



Gambar 1.4 Flowchart pembatalan pesanan

F. Algoritma Sistem

1. Data Awal

Dalam sistem program kami, terdapat data awal yang terdiri dari nama makanan/minuman, harga makanan/minuman, dan stok makanan/minuman. Data dibagi menjadi 6 set array berbeda, sesuai dengan jenis makanan dan minumannya.

```
#VARIABEL DATA MENU --> Berisi variabel dan array tentang nama, harga, dan stok produk yang dijual
#=====
sushi = 20
list_Sushi = ["Spicy Salmon Sushi", "Black Pepper Tuna Sushi", "Chicken Nanban Roll", "Maguro Tataki", "Chikuwa Cheese Roll", "Salmon Tempura Floss Roll",
              "Kani Mayo Mentai Roll", "Oase Roll", "Veggie Roll", "Crispy Unagi Roll", "Tamago Maki", "Tuna Salad Maki",
              "Beef Tamago Cheese Maki", "Corn Cheese Maki", "Kani Mentai Sushi", "Salmon Mentai Sushi", "Unagi Sushi", "Tamago Sushi",
              "Tamago Sushi", "Salmon Cheese Roll"]
stok_sushi = [100 for i in range(sushi)]
harga_Sushi = [27500, 16500, 44000, 27500, 44000, 44000, 33000, 33000, 33000, 22000, 55000, 22000, 33000, 55000, 11000,
              16500, 27500, 27500, 11000, 55000]

ramen = 5
list_Ramen = ["Shoyu Ramen", "Spicy Miso Ramen", "Grilled Chicken Ramen", "Chicken Katsu Ramen", "Goma Kara Ramen"]
stok_ ramen = [100 for i in range(ramen)]
harga_Ramen = [22000, 22000, 48000, 50000, 49500]

ricebowl = 11
list_Ricebowl = ["Chicken Karaage Don", "Crispy Salmon Mentai Don", "Yakiniku Don", "Chicken Teriyaki Don", "Spicy Ten Don",
                "Chicken Karaage Mentai Rice", "Beef Teriyaki Mentai Rice", "Salmon Karaage Mentai Rice", "Cheese Katsu Don",
                "Chicken Nanban Don", "Salmon Tartar Don"]
stok_ricebowl = [100 for i in range(ricebowl)]
harga_Ricebowl = [22000, 27500, 27500, 22000, 27500, 27500, 27500, 27500, 22000, 27500, 27500]

minuman = 7
list_Minuman = ["Ocha", "Ice Tea", "Lemon Tea", "Iced Sweet Lychee Tea", "Iced Mango Tea", "Iced Passion Fruit Tea", "Lemongrass Lychee Tea"]
stok_minuman = [100 for i in range(minuman)]
harga_Minuman = [8000, 7000, 10000, 25000, 25000, 25000, 20000]

PaketWibu = 2
list_PaketWibu = ["PaketWibu A", "PaketWibu B"]
stok_PaketWibu = [100 for i in range(PaketWibu)]
harga_PaketWibu = [50000, 55000]

Appetizer = 6
list_Appetizer = ["Cheese Dorayaki", "Chocolate Dorayaki", "Strawberry Choux", "Mix Dorayaki", "Matcha Choux", "Chocolate Choux"]
stok_Appetizer = [100 for i in range(Appetizer)]
harga_Appetizer = [16500, 16500, 17000, 16500, 17000, 17000]
#=====
```

Gambar 2.1.1 Potongan kode yang menyimpan variabel data menu

2. Algoritma Level 1 (Tampilan Depan)

Dalam awal penggunaan program, akan muncul tulisan selamat datang yang diikuti dengan pertanyaan “y” / ”n” untuk pesan atau tidak pesan.

Jika pengguna menjawab “n” maka program akan mengulang dari awal lagi. Namun, jika menjawab “y” akan ditanyakan nama pengguna dan pilihan pengguna untuk dine-in atau take-out. Jika pengguna memilih dine-in maka akan diminta input nomor meja pengguna.

```
-----
|                               |
|       Welcome to Computational Sushi       |
|                               |
|       Restoran kami menyediakan berbagai menu asli dari Jepang |
|       dengan kualitas bahan yang paling tinggi. |
|                               |
|-----|
Mulai pesan? (y/n): y
0---CUSTOMER INFORMATION---
Masukkan nama kamu: irfan
Dine In / Take Away? (1/2): 1
Masukkan nomor meja yang kamu duduki: 100
```

Gambar 2.2.1 Tulisan selamat datang beserta tahap input informasi awal pengguna

Berikut adalah potongan kode yang kami gunakan di algoritma ini:

```
110 #ALGORITMA PROGRAM
111 while True:
112     mulai_pesan = False
113     while (mulai_pesan == False):
114
115         #TAMPILAN DEPAN
116         print("                               ")
117         print("       Welcome to Computational Sushi       ")
118         print("                               ")
119         print("       Restoran kami menyediakan berbagai menu asli dari Jepang ")
120         print("       dengan kualitas bahan yang paling tinggi. ")
121         print("                               ")
122         print("                               ")
123         jadi_pesan = input("Mulai pesan? (y/n): ")
124         if (jadi_pesan == "y"):
125             mulai_pesan = True
126             time.sleep(0.3)
127             os.system('cls')
128         else:
129             time.sleep(0.3)
130             os.system('cls')
```

Gambar 2.2.2 Potongan kode yang menampilkan tampilan depan dari program

```

133 #ISI IDENTITAS
134 Nama_pemesan = ""
135 while Nama_pemesan == "":
136     print("---CUSTOMER INFORMATION---")
137     Nama_pemesan = input("Masukkan nama kamu: ")
138     if (Nama_pemesan != ""):
139         break
140     else:
141         os.system('cls')
142
143 #Pemilihan Dine In/Take Away
144 Opsi = 0
145 while Opsi != 1 or Opsi != 2:
146
147     Opsi = input("Dine In / Take Away? (1/2): ")
148     if (Opsi != ""):
149         Opsi = int(Opsi)
150         if Opsi == 1:
151             Nomor_meja = 0
152             while True:
153                 Nomor_meja = input("Masukkan nomor meja yang kamu duduki: ")
154                 if (Nomor_meja != ""):
155                     Nomor_meja = int(Nomor_meja)
156                     if Nomor_meja <= max_meja:
157                         break
158                     else:
159                         print("Pastikan nomor meja yang Anda masukkan benar dan sesuai")
160             break
161         elif Opsi == 2:
162             break
163
164     time.sleep(0.3)
165     os.system('cls')

```

Gambar 2.2.3 Potongan kode algoritma pengisian identitas

3. Algoritma Level 2 (Pesanan)

Setelah identitas dan pilihan dine-in atau take-out telah selesai diisi, akan muncul daftar jenis menu yang dapat dipilih dengan memasukkan nomor sesuai pada pilihannya.

```

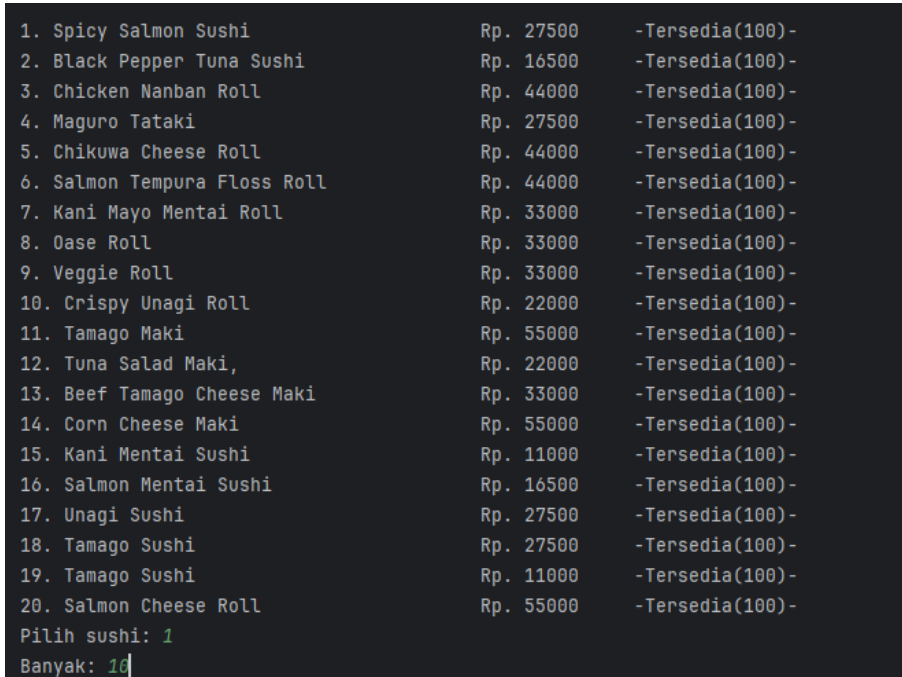
0  -----
|Daftar Jenis Pesanan: |
|1. Sushi              |
|2. Ramen              |
|3. Rice Bowl         |
|4. Minuman            |
|5. Paket Wibu        |
|6. Appetizer         |
|-----|
|7. Cek Pesanan       |
|8. Batal Pesan       |
|-----|
Opsi: 1

```

Gambar 2.3.1 Tampilan daftar jenis menu dan input pilihan

Jika pengguna input pilihannya maka akan muncul menu sesuai jenis menu yang dipilih. Menu ini berisikan nomor, nama makanan atau minuman, harga, dan stok. Pengguna dapat memilih makanan atau minuman dengan input nomor yang sesuai, lalu memasukkan jumlah

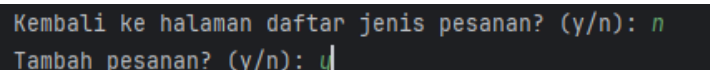
atau banyak yang diinginkan. Algoritma yang digunakan untuk masing-masing makanan atau minuman sama, hanya berbeda pada variabel untuk masing-masing makanan atau minuman.



1. Spicy Salmon Sushi	Rp. 27500	-Tersedia(100)-
2. Black Pepper Tuna Sushi	Rp. 16500	-Tersedia(100)-
3. Chicken Nanban Roll	Rp. 44000	-Tersedia(100)-
4. Maguro Tataki	Rp. 27500	-Tersedia(100)-
5. Chikuwa Cheese Roll	Rp. 44000	-Tersedia(100)-
6. Salmon Tempura Floss Roll	Rp. 44000	-Tersedia(100)-
7. Kani Mayo Mentai Roll	Rp. 33000	-Tersedia(100)-
8. Oase Roll	Rp. 33000	-Tersedia(100)-
9. Veggie Roll	Rp. 33000	-Tersedia(100)-
10. Crispy Unagi Roll	Rp. 22000	-Tersedia(100)-
11. Tamago Maki	Rp. 55000	-Tersedia(100)-
12. Tuna Salad Maki,	Rp. 22000	-Tersedia(100)-
13. Beef Tamago Cheese Maki	Rp. 33000	-Tersedia(100)-
14. Corn Cheese Maki	Rp. 55000	-Tersedia(100)-
15. Kani Mentai Sushi	Rp. 11000	-Tersedia(100)-
16. Salmon Mentai Sushi	Rp. 16500	-Tersedia(100)-
17. Unagi Sushi	Rp. 27500	-Tersedia(100)-
18. Tamago Sushi	Rp. 27500	-Tersedia(100)-
19. Tamago Sushi	Rp. 11000	-Tersedia(100)-
20. Salmon Cheese Roll	Rp. 55000	-Tersedia(100)-
Pilih sushi: 1		
Banyak: 10		

Gambar 2.3.2 Tampilan daftar jenis makanan dan minuman

Setelah itu akan muncul pertanyaan “y”/”n” untuk kembali ke daftar jenis menu. Jika input “y” maka pengguna akan kembali ke daftar jenis menu. Namun jika menjawab “n” maka pengguna akan diberi pertanyaan “y”/”n” apakah akan memesan lagi atau tidak. Jika menjawab “n” pengguna akan kembali ke daftar jenis menu, namun jika menjawab “y” pengguna akan kembali ke daftar jenis makanan atau minuman sesuai dengan jenis menu yang sudah dipilih tadi.



```
Kembali ke halaman daftar jenis pesanan? (y/n): n
Tambah pesanan? (y/n): y
```

Gambar 2.3.3 Tampilan input lanjutan setelah pesan

Algoritma ini akan mengulang terus-menerus sampai pengguna memilih opsi selain nomor 1 sampai 6 di menu daftar jenis makanan atau minuman, yaitu pilihan cek pesanan, atau batal pesan.

Berikut adalah potongan kode yang digunakan untuk algoritma ini:

```

indeks_pesanan = 0
choice_jenis = 0
while (mulai_pesanan == True):
    #LIST JENIS MAKANAN
    print("_____")
    print("|Daftar Jenis Pesanan: |")
    print("|1. Sushi |")
    print("|2. Ramen |")
    print("|3. Rice Bowl |")
    print("|4. Minuman |")
    print("|5. Paket Wibu |")
    print("|6. Appetizer |")
    print("|_____")
    print("|7. Cek Pesanan |")
    print("|8. Batal Pesan |")
    print("|_____")

```

```

while True:
    choice_jenis = input("Opsi: ")
    if (choice_jenis != ""):
        choice_jenis = int(choice_jenis)
        time.sleep(0.5)
        os.system('cls')
        break

```

Gambar 2.3.4 Potongan kode algoritma tampilan list jenis makanan

```

#Pemilihan Sushi
if (choice_jenis == 1):
    pesan_lagi = True
    while pesan_lagi:
        print('Masukkan angka yang sesuai dengan nomor dipilihan.')
        print("Pilihan Sushi:")
        print()

        #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}-")

        #Memilih Sushi
        while True:
            pilihan = input("Pilih sushi: ")
            if pilihan != "":
                pilihan = int(pilihan)
                if pilihan > 0 and pilihan < len(list_Sushi)+1:
                    break
                else:
                    print('Pastikan angka yang Anda masukkan benar.')

```

Gambar 2.3.5 Potongan kode algoritma pemilihan, pengecekan stok, dan memilih makanan atau minuman (sampel yang diambil “Sushi”)


```

#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()

#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

```

Gambar 2.3.6 Potongan kode algoritma penentuan banyak, pencatatan, dan pengecekan makanan atau minuman (sampel yang diambil “Sushi”)

```

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

#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 (nambah_pesanan == 'n'):
            pesan_lagi = False
        elif (nambah_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

```

Gambar 2.3.7 Potongan kode algoritma pengurangan stok dan opsi kembali ke halaman utama/tambah pesanan (sampel yang diambil “Sushi”)

4. Algoritma Level 3 (Cek Pesanan)

Saat pengguna menginput angka 7 di menu daftar jenis makanan atau minuman, akan muncul pesan saat ini yang sudah ditambahkan ataupun belum oleh pemesan. Apabila pengguna belum melakukan pesanan apa-apa, program otomatis akan dialihkan ke halaman daftar jenis pesanan.

```
Maaf kamu belum melakukan pesanan, kami akan alihkan ke daftar jenis pesanan. . .
```

Gambar 2.4.1 Tampilan jika pengguna tidak memesan

Jika sudah melakukan pesanan, akan muncul opsi “Kurangi pesanan”, “Kembali ke halaman daftar jenis pesanan”, dan “Checkout”. Pengguna diminta memilih sesuai nomor pada opsi tersebut. Jika pengguna memilih opsi “Kembali ke halaman daftar jenis pesanan”, maka program akan menampilkan daftar jenis pesanan. Namun, jika pengguna memilih opsi “Kurangi pesanan”, maka pengguna akan diminta untuk memilih nama dan banyak pesanan yang ingin dikurangi dan program akan mengembalikan stok dari pesanan yang sudah dikurangi, selanjutnya akan muncul opsi “Kembali ke halaman daftar jenis pesanan”, “Kurangi pesanan”, atau “Checkout” kembali sampai pengguna memilih opsi untuk kembali ke halaman daftar jenis pesanan.

```
Opsi: 7
Pesanan saat ini:
1. Spicy Miso Ramen: 2

Opsi:
1. Kurangi pesanan
2. Kembali ke halaman daftar jenis pesanan
3. Checkout
Pilih opsi: 1

Pilih pesanan yang mau dikurangi: 1
Banyak pesanan yang dikurang: |
```

Gambar 2.4.2 Tampilan cek pesanan dan pilihan kurangi pesanan

Berikut adalah potongan kode yang digunakan untuk algoritma ini:

```

#Cek Pesanan
elif (choice_jenis == 7):
    if indeks_pesanan != 0:

        cek_pesanan = True

        #Menampilkan pesanan saat ini dan memberi opsi Kembali / Kurangi pesanan
        while cek_pesanan:
            print("Pesanan saat ini: ")
            if indeks_pesanan != 0:
                for i in range(indeks_pesanan):
                    print(f"{i+1}. {pesanan_nama[i]}: {pesanan_jumlah[i]}")
            else:
                print("-")
            print()
            print("Opsi: ")
            print("1. Kurangi pesanan")
            print("2. Kembali ke halaman daftar jenis pesanan")
            print("3. Checkout")

            while True:
                choice_cek_pesanan = input("Pilih opsi: ")
                if choice_cek_pesanan != "":
                    choice_cek_pesanan = int(choice_cek_pesanan)
                    break
            print()

            #Kembali ke halaman Daftar Jenis Makanan
            if choice_cek_pesanan == 2:
                print("Kembali ke halaman Daftar Jenis Pesanan. . .")
                time.sleep(1)
                os.system('cls')
                cek_pesanan = False
                pesan_lagi = False

```

Gambar 2.4.3 Potongan kode algoritma menampilkan pesanan saat ini dan pemberian opsi untuk pengguna

```

#Mengurangi pesanan
elif choice_cek_pesanan == 1:
    if indeks_pesanan != 0:
        while True:
            pilihan_kurang = input("Pilih pesanan yang mau dikurangi: ")
            if pilihan_kurang != "":
                pilihan_kurang = int(pilihan_kurang)
                if pilihan_kurang > 0 and pilihan_kurang <= indeks_pesanan:
                    break
            else:
                print("Masukkan angka yang sesuai")

        while True:
            banyak_berkurang = input("Banyak pesanan yang dikurang: ")
            if banyak_berkurang != "":
                banyak_berkurang = int(banyak_berkurang)
                if banyak_berkurang > 0 and banyak_berkurang <= pesanan_jumlah[pilihan_kurang-1]:
                    time.sleep(0.5)
                    os.system('cls')
                    break
            else:
                print("Masukkan banyak yang sesuai")

        item_kurang = pesanan_nama[pilihan_kurang-1]
        for i in range(indeks_pesanan):
            if item_kurang == pesanan_nama[i]:
                if pesanan_jumlah[i] > 0:
                    pesanan_jumlah[i] -= banyak_berkurang
                    pesanan_total[i] -= banyak_berkurang * pesanan_harga[i]
                    if pesanan_jumlah[i] <= 0:
                        pesanan_nama[i] = ""
                        pesanan_harga[i] = 0
                        pesanan_total[i] = 0
                        for j in range(i, indeks_pesanan+1):
                            pesanan_nama[j] = pesanan_nama[j+1]
                            pesanan_harga[j] = pesanan_harga[j+1]
                            pesanan_jumlah[j] = pesanan_jumlah[j+1]
                            pesanan_total[j] = pesanan_total[j+1]
                        indeks_pesanan -= 1

```

Gambar 2.4.4 Potongan kode algoritma mengurangi pesanan

```

#Mengembalikan stok apabila ada menu yang dikurangi
for i in range (sushi):
    if list_Sushi[i]==item_kurang:
        stok_sushi[i]+=banyak_berkurang
for i in range (ramen):
    if list_Ramen[i]==item_kurang:
        stok_ramen[i]+=banyak_berkurang
for i in range (ricebowl):
    if list_Ricebowl[i]==item_kurang:
        stok_ricebowl[i]+=banyak_berkurang
for i in range (minuman):
    if list_Minuman[i]==item_kurang:
        stok_minuman[i]+=banyak_berkurang
for i in range (PaketWibu):
    if list_PaketWibu[i]==item_kurang:
        stok_PaketWibu[i]+=banyak_berkurang
for i in range (Appetizer):
    if list_Appetizer[i]==item_kurang:
        stok_Appetizer[i]+=banyak_berkurang

```

Gambar 2.4.5 Potongan kode algoritma mengembalikan stok apabila ada menu yang dikurangi

5. Algoritma Level 4 (Batal Pesan)

Saat pengguna menginput angka 8 di menu daftar jenis makanan atau minuman, pengguna diminta untuk mengkonfirmasi pembatalan pesanan. Jika pengguna setuju membatalkan pesanan, program akan kembali ke halaman tampilan depan dan pesanan yang dipesan akan dikembalikan. Namun, jika pengguna tidak jadi membatalkan pesanan, program akan kembali ke halaman daftar jenis pesanan.

```

-----
|Daftar Jenis Pesanan: |
|1. Sushi              |
|2. Ramen              |
|3. Rice Bowl         |
|4. Minuman            |
|5. Paket Wibu        |
|6. Appetizer          |
|-----|
|7. Cek Pesanan       |
|8. Batal Pesan       |
|-----|
Opsi: 8
Apakah yakin untuk membatalkan pesanan? (y/n): y

```

Gambar 2.5.1 Tampilan Batal Pesan

Berikut adalah potongan kode yang digunakan untuk algoritma ini:

```

#Batal Pesan
elif (choice_jenis == 8):
    while True:
        yakin_batal = input("Apakah yakin untuk membatalkan pesanan? (y/n): ")
        if yakin_batal == 'y':
            print("Pesanan dibatalkan, beralih ke halaman utama. . .")
            time.sleep(1.5)
            os.system('cls')
            mulai_pesan = False

            #Mengembalikan stok apabila pesanan dibatalkan
            for i in range (indeks_pesanan):
                for j in range (sushi):
                    if list_Sushi[j]==pesanan_nama[i]:
                        stok_sushi[j]+=pesanan_jumlah[i]
                for j in range (ramen):
                    if list_Ramen[j]==pesanan_nama[i]:
                        stok_ramen[j]+=pesanan_jumlah[i]
                for j in range (ricebowl):
                    if list_Ricebowl[j]==pesanan_nama[i]:
                        stok_ricebowl[j]+=pesanan_jumlah[i]
                for j in range (minuman):
                    if list_Minuman[j]==pesanan_nama[i]:
                        stok_minuman[j]+=pesanan_jumlah[i]
                for j in range (PaketWibu):
                    if list_PaketWibu[j]==pesanan_nama[i]:
                        stok_PaketWibu[j]+=pesanan_jumlah[i]
                for j in range (Appetizer):
                    if list_Appetizer[j]==pesanan_nama[i]:
                        stok_Appetizer[j]+=pesanan_jumlah[i]
            break
        elif yakin_batal == 'n':
            time.sleep(0.5)
            os.system('cls')
            pesan_lagi = False
            break

```

Gambar 2.5.2 Potongan kode algoritma batal pesan dan pengembalian stok apabila pesanan dibatalkan

6. Algoritma Level 5 (Checkout)

Saat pengguna berada pada halaman cek pesanan dan menginput angka 3, akan muncul konfirmasi untuk melakukan checkout. Jika konfirmasi checkout “n”, akan dialihkan ke halaman cek pesanan kembali. Jika konfirmasi checkout “y”, akan muncul pilihan opsi pembayaran dengan 4 opsi, yaitu tunai, atm, kredit, dan Q-RIS.

```

Opsi pembayaran:
1. Tunai
2. ATM
3. Kredit
4. QRIS
Pilih pembayaran: 

```

Gambar 2.6.1 Tampilan opsi pembayaran

Pengguna diminta memilih sesuai nomor pada opsi tersebut. Jika pengguna memilih opsi tunai, atm atau kredit, maka akan di print struk pembelian dan pengguna diminta untuk melakukan pembayaran di kasir. Namun, jika pengguna memilih opsi pembayaran Q-RIS, maka di struk pembelian akan diberikan QR Q-RIS yang bisa dipindai pengguna dan selanjutnya pengguna akan diminta untuk mengkonfirmasi pembayaran. Jika konfirmasi pembayaran ‘y’, program akan kembali ke halaman tampilan depan. Jika konfirmasi pembayaran ‘n’, program akan kembali menampilkan struk pembayaran sampai konfirmasi pembayaran ‘y’.

```

#Checkout
elif choice_cek_pesanan == 3:
    if indeks_pesanan != 0:
        #HARGA AKHIR
        total_harga = 0
        for i in range(max_pesanan):
            total_harga += pesanan_total[i]

        #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')

                #OPSI PEMBAYARAN
                opsi_pembayaran = ["Tunai", "ATM", "Kredit", "QRIS"]
                print("Opsi pembayaran: ")
                for i in range(4):
                    print(f"{i+1}. {opsi_pembayaran[i]}")

                while True:
                    choice_bayar = input("Pilih pembayaran: ")
                    if choice_bayar != "":
                        choice_bayar = int(choice_bayar)
                        if (choice_bayar <= 0 or choice_bayar > 4):
                            print("Pastikan pilihan Anda sesuai.")
                        else:
                            break

                #PENCETAKAN STRUK PESANAN
                print("Struk pesananmu sedang dicetak. . . ")
                time.sleep(3)
                os.system('cls')

```

Gambar 2.6.2 Potongan kode algoritma konfirmasi checkout dan pemilihan opsi pembayaran

```

#STRUK PESANAN
if choice_bayar != 4: #Selain QRIS
    print("_____")
    print("STRUK PESANAN")
    print()
    print(f>Nama: {Nama_pemesan}")
    print(f"Nomor Antrian: {antrian}")
    print("-----")
    if (Ops == 1):
        print("DINE IN")
        print(f"Nomor Meja: {Nomor_meja}")
    else:
        print("TAKE AWAY")
    print("-----")
    for i in range(indeks_pesanan):
        print(f"{pesanan_nama[i]: {pesanan_jumlah[i]} x Rp.{pesanan_harga[i]} : Rp.{pesanan_total[i]})")
    print(f"Total Harga: Rp{total_harga}")
    print(f"Pembayaran: {opsi_pembayaran[choice_bayar-1]}")
    print("Silakan lakukan pembayaran di KASIR, terima kasih.")
    print("_____")

```

Gambar 2.6.3 Potongan kode algoritma pencetakan struk pesanan (sampel yang diambil pembayaran selain Q-RIS)

```

#Konfirmasi Pembayaran
while True:
    konfirmasi_pembayaran = input("Konfirmasi pembayaran (y/n): ")
    if konfirmasi_pembayaran != "":
        break

    if konfirmasi_pembayaran == 'y':
        print("Terima kasih sudah melakukan pembayaran, pesanan segera disiapkan, mohon ditunggu.")
        time.sleep(2)
        os.system('cls')
        break
    elif konfirmasi_pembayaran == 'n':
        print("Maaf pembayaran belum berhasil. Silakan coba lagi.")
        time.sleep(2)
        os.system('cls')

```

Gambar 2.6.4 Potongan kode algoritma konfirmasi pembayaran

G. Deliverable

1. Source code

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

2. Video Presentasi Tugas

https://itbdsti-my.sharepoint.com/:v:/g/personal/19624088_mahasiswa_itb_ac_id/ERtOs-xANQFFudOLkCbM30EBnkIuF5MTCCtailLwRxdCbQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAIoiJPbmVEcm12ZUZvckJlc2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=yKqpbu

BAB III

KESIMPULAN

Kami merancang suatu sistem pemesanan makanan di restoran yang memungkinkan pelanggan untuk melihat serta memilih hidangan yang diinginkan, menambahkannya ke dalam daftar pesanan, dan memodifikasi pilihan mereka dengan cepat seperti menambah atau mengurangi jumlah pesanan sesuai keinginan. Dengan menggunakan sistem ini, pemesanan makanan pada suatu restoran menjadi lebih efektif dan efisien.

Dengan menggunakan teknik berpikir komputasional, kami berhasil mengupas mekanisme di balik sistem pemesanan makanan. Dengan dekomposisi, kami mampu memecah urutan kerja dari sistem menjadi versi yang paling sederhana untuk diselesaikan. Dengan pengenalan pola, kami dapat mengenali pola-pola yang ada pada sistem dan memanfaatkannya agar hasil algoritma kami lebih efisien. Dengan abstraksi, kami bisa menyimpulkan cara kerja sistem dengan memilah antara informasi yang penting dan tidak penting. Dengan algoritma, kami membuat langkah-langkah dan instruksi yang berhasil menyelesaikan pekerjaan. Dengan semua teknik ini, kami berhasil membuat replika simulasi *Food Ordering System* sederhana yang menangkap inti dari sistem tersebut yang sesuai dengan kegunaannya di dunia nyata.

DAFTAR PUSTAKA

- Barr, V., & Stephenson, C. (2011). *Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community?* ACM Inroads, 2(1), 48-54.
- Grover, S., & Pea, R. (2013). *Computational Thinking in K-12: A Review of the State of the Field*. Educational Researcher, 42(1), 38-43.
- Selby, C., & Woollard, J. (2014). *Computational Thinking: The Developing Definition*. University of Southampton.
- Wing, J. M. (2006). *Computational Thinking*. Communications of the ACM, 49(3), 33-35.
- Wing, J. M. (2011). *Research Notebook: Computational Thinking—What and Why?* The Link Magazine, Carnegie Mellon University.