PRACTICUM REPORT
ALGORITHM AND DATA STRUCTURES
MODUL 4 : SEARCHING

Disusun Oleh :

ONIC AGUSTINO

L200234275

X

INFORMATICS ENGINEERING

FACULTY OF COMMUNICATION AND INFORMATICS

UNIVERSITAS MUHAMMADIYAH SURAKARTA

YEARS 2024/2025

1.11 Questions

1. Create a search function that, instead of returning True/False, returns all location indexes of the elements searched. So, for example, in the student list on page 49 we look for students who come from Klaten, we will get [6, 8]. If what you are looking for is not found, this function will return an empty list.

Module_4 > Question > 1.py

```python
def cariSemuaIndex(x, data):
    """
    Mencari semua index yang mengandung nilai x dalam data.
    Mengembalikan list berisi index-index tersebut.
    Jika tidak ditemukan, mengembalikan list kosong.
    """
    hasil = []
    for i in range(len(data)):
        if data[i] == x:
            hasil.append(i)
    return hasil

# Contoh penggunaan
Daftar = [
    ["Ayu", "Surakarta"],
    ["Bagyo", "Sragen"],
    ["Chandra", "Surakarta"],
    ["Dini", "Surakarta"],
    ["Ela", "Boyolali"],
    ["Fani", "Salatiga"],
    ["Gilang", "Klaten"],
    ["Heru", "Karanganyar"],
    ["Ika", "Klaten"],
    ["Joko", "Boyolali"]
]

def cariKota(kota, data):
    """
    Mencari semua index mahasiswa yang berasal dari kota tertentu.
    """
    return cariSemuaIndex(kota, [mhs[1] for mhs in data])

# Test fungsi
print(cariKota("Klaten", Daftar))
print(cariKota("Jakarta", Daftar))
print(cariKota("Surakarta", Daftar))
```

Picture 1.1 the code 1.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/1.py"
[6, 8]
[]
[0, 2, 3]
```

Picture 1.2 the output

2. From the list of students above, create a function to find the smallest pocket money among them.

Module_4 > Question > 2.py > ...

```python
def cariUangSakuTerkecil(daftar):
    """
    Mencari uang saku terkecil dari daftar mahasiswa.
    Mengasumsikan setiap mahasiswa memiliki data: [nama, kota, uang_saku]
    """
    if not daftar:  # Jika daftar kosong
        return None

    # Inisialisasi dengan uang saku mahasiswa pertama
    terkecil = daftar[0][2]  # Mengambil uang saku dari data pertama

    # Mencari uang saku terkecil
    for mahasiswa in daftar:
        if mahasiswa[2] < terkecil:
            terkecil = mahasiswa[2]

    return terkecil

# Contoh penggunaan
Daftar = [
    ["Ayu", "Surakarta", 250000],
    ["Bagyo", "Sragen", 275000],
    ["Oky", "Klaten", 20000],
    ["Chandra", "Surakarta", 235000],
    ["Dini", "Surakarta", 240000],
    ["Ela", "Boyolali", 245000],
    ["Fani", "Salatiga", 245000],
    ["Gilang", "Klaten", 245000],
    ["Heru", "Karanganyar", 265000],
    ["Ika", "Klaten", 275000],
    ["Joko", "Boyolali", 260000],
    ["Naufal", "Klaten", 10000]
]

# Test fungsi
uang_terkecil = cariUangSakuTerkecil(Daftar)
print(f"Uang saku terkecil adalah: Rp {uang_terkecil}")
```

Picture 2.1 the code 2.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/2.py"
Uang saku terkecil adalah: Rp 10000
```

Picture 2.2 the output.

3. Change the program above to return the student object that has the smallest allowance. If there is more than one student whose pocket money is the smallest, all student objects are returned.

Module_4 > Question > 3.py > cariMahasiswaUangSakuTerkecil

```python
def cariMahasiswaUangSakuTerkecil(daftar):
    """
    Mencari mahasiswa dengan uang saku terkecil dari daftar mahasiswa.
    Jika ada lebih dari satu mahasiswa dengan uang saku terkecil, semua dikembalikan.
    """
    if not daftar:  # Jika daftar kosong
        return []

    # Cari uang saku terkecil
    terkecil = daftar[0][2]
    for mahasiswa in daftar:
        if mahasiswa[2] < terkecil:
            terkecil = mahasiswa[2]

    # Cari semua mahasiswa dengan uang saku terkecil
    hasil = [mahasiswa for mahasiswa in daftar if mahasiswa[2] == terkecil]
    return hasil

# Contoh penggunaan
Daftar = [
    ["Ayu", "Surakarta", 250000],
    ["Bagyo", "Sragen", 275000],
    ["Oky", "Klaten", 20000],
    ["Chandra", "Surakarta", 235000],
    ["Dini", "Surakarta", 240000],
    ["Ela", "Boyolali", 245000],
    ["Fani", "Salatiga", 245000],
    ["Gilang", "Klaten", 245000],
    ["Heru", "Karanganyar", 265000],
    ["Ika", "Klaten", 275000],
    ["Joko", "Boyolali", 260000],
    ["Naufal", "Klaten", 20000]
]

# Test fungsi
mahasiswa_terkecil = cariMahasiswaUangSakuTerkecil(Daftar)
print("Mahasiswa dengan uang saku terkecil:")
for mhs in mahasiswa_terkecil:
    print(mhs)
```

Picture 3.1 the code 3.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/3.py"
Mahasiswa dengan uang saku terkecil:
['Oky', 'Klaten', 20000]
['Naufal', 'Klaten', 20000]
```

Picture 3.2 the output.

4. Create a function that returns all student objects whose allowance is less than 250000.

Module_4 > Question > 4.py > ...

```python
def cariMahasiswaUangSakuKurang(daftar, batas):
    """
    Mencari semua mahasiswa dengan uang saku kurang dari batas tertentu.
    """
    hasil = [mahasiswa for mahasiswa in daftar if mahasiswa[2] < batas]
    return hasil

# Contoh penggunaan
Daftar = [
    ["Ayu", "Surakarta", 250000],
    ["Bagyo", "Sragen", 275000],
    ["Oky", "Klaten", 20000],
    ["Chandra", "Surakarta", 235000],
    ["Dini", "Surakarta", 240000],
    ["Ela", "Boyolali", 245000],
    ["Fani", "Salatiga", 245000],
    ["Gilang", "Klaten", 245000],
    ["Heru", "Karanganyar", 265000],
    ["Ika", "Klaten", 275000],
    ["Joko", "Boyolali", 260000],
    ["Naufal", "Klaten", 20000]
]

# Test fungsi
mahasiswa_kurang = cariMahasiswaUangSakuKurang(Daftar, 250000)
print("Mahasiswa dengan uang saku kurang dari 250000:")
for mhs in mahasiswa_kurang:
    print(mhs)
```

Picture 4.1 the code 4.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/4.py"
Mahasiswa dengan uang saku kurang dari 250000:
['Oky', 'Klaten', 20000]
['Chandra', 'Surakarta', 235000]
['Dini', 'Surakarta', 240000]
['Ela', 'Boyolali', 245000]
['Fani', 'Salatiga', 245000]
['Gilang', 'Klaten', 245000]
['Naufal', 'Klaten', 20000]
```

Picture 4.2 the output.

5. Write a program to search for an item in a linked list.

Module_4 > Question > 5.py > ...

```python
class Node:
    def __init__(self, data=None):
        self.data = data
        self.next = None

class LinkedList:
    def __init__(self):
        self.head = None

    def tambah(self, data):
        """
        Menambahkan elemen baru ke akhir linked list.
        """
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        current = self.head
        while current.next:
            current = current.next
        current.next = new_node

    def cari(self, item):
        """
        Mencari item dalam linked list.
        Mengembalikan True jika ditemukan, False jika tidak.
        """
        current = self.head
        while current:
            if current.data == item:
                return True
            current = current.next
        return False

    def cetak(self):
        """
        Mencetak semua elemen dalam linked list.
        """
        current = self.head
        while current:
            print(current.data, end=" -> ")
            current = current.next
```

```
43              print("None")
44
45      # Contoh penggunaan
46      ll = LinkedList()
47      ll.tambah(10)
48      ll.tambah(20)
49      ll.tambah(30)
50      ll.tambah(40)
51
52      print("Isi Linked List:")
53      ll.cetak()
54
55      # Mencari item
56      item = 30
57      if ll.cari(item):
58          print(f"Item {item} ditemukan dalam linked list.")
59      else:
60          print(f"Item {item} tidak ditemukan dalam linked list.")
61
62      item = 50
63      if ll.cari(item):
64          print(f"Item {item} ditemukan dalam linked list.")
65      else:
66          print(f"Item {item} tidak ditemukan dalam linked list.")
```

Picture 5.1 the code 5.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/5.py"
Isi Linked List:
10 -> 20 -> 30 -> 40 -> None
Item 30 ditemukan dalam linked list.
Item 50 tidak ditemukan dalam linked list.
```

Picture 5.2 the output.

6. Binary search. Change the function binSe on page 54 to return the index of the location of the element found. If you don't find it, you will return it False.

Module_4 > Question > 6.py > ...

```python
def binSe(arr, target):
    """
    Melakukan binary search pada array yang sudah terurut.
    Mengembalikan index elemen jika ditemukan, atau False jika tidak ditemukan.
    """
    low = 0
    high = len(arr) - 1

    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == target:
            return mid  # Mengembalikan index elemen yang ditemukan
        elif arr[mid] < target:
            low = mid + 1
        else:
            high = mid - 1

    return False  # Mengembalikan False jika elemen tidak ditemukan

# Contoh penggunaan
arr = [10, 20, 30, 40, 50]
target = 30
result = binSe(arr, target)
if result is not False:
    print(f"Elemen {target} ditemukan pada index {result}.")
else:
    print(f"Elemen {target} tidak ditemukan.")

target = 25
result = binSe(arr, target)
if result is not False:
    print(f"Elemen {target} ditemukan pada index {result}.")
else:
    print(f"Elemen {target} tidak ditemukan.")
```

Picture 6.1 the code 6.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/6.py"
Elemen 30 ditemukan pada index 2.
Elemen 25 tidak ditemukan.
```

Picture 6.2 the output.

7. Binary search. Change the function binSe to return all the location indexes of the elements found. Example: looking for the number 6 in the list [2, 3, 5, 6, 6, 6, 8, 9, 9, 10,11, 12, 13, 13, 14] will return [3, 4, 5]. Because it's already sorted, "just look at the left and right".

Module_4 > Question > 7.py > ...

```python
def binSeAll(arr, target):
    """
    Melakukan binary search pada array yang sudah terurut.Mengembalikan semua index elemen yang ditemukan dalam bentuk list.
    """
    low = 0
    high = len(arr) - 1
    result = []

    # Binary search to find one occurrence of the target
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == target:
            # Expand to the left and right to find all occurrences
            left = mid
            while left >= 0 and arr[left] == target:
                result.append(left)
                left -= 1
            right = mid + 1
            while right < len(arr) and arr[right] == target:
                result.append(right)
                right += 1
            result.sort()  # Ensure the indexes are sorted
            return result
        elif arr[mid] < target:
            low = mid + 1
        else:
            high = mid - 1
    return []  # Return an empty list if the target is not found
# Contoh penggunaan
arr = [2, 3, 5, 6, 6, 6, 8, 9, 9, 10, 11, 12, 13, 13, 14]
target = 6
result = binSeAll(arr, target)
if result:
    print(f"Elemen {target} ditemukan pada index {result}.")
else:
    print(f"Elemen {target} tidak ditemukan.")
target = 7
result = binSeAll(arr, target)
if result:
    print(f"Elemen {target} ditemukan pada index {result}.")
else:
    print(f"Elemen {target} tidak ditemukan.")
```

Picture 7.1 the code 7.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/7.py"
Elemen 6 ditemukan pada index [3, 4, 5].
Elemen 7 tidak ditemukan.
```

Picture 7.2 the output.

8. In the number guessing game that you created in Chapter 1 (questions number 12, page 19), If the number to be guessed is between 1 and 100, the maximum number of guesses should be 7. If it is between 1 and 1000, the maximum number of guesses is 10. Why is that? What's the pattern?

Module_4 > Question > 8.py > ...

```python
import math
import random

def tebak_angka():
    print("Permainan Tebak Angka")

    # Memilih rentang angka
    print("Pilih rentang angka:")
    print("1. 1 sampai 100")
    print("2. 1 sampai 1000")
    pilihan = int(input("Masukkan pilihan (1/2): "))

    if pilihan == 1:
        batas_atas = 100
    elif pilihan == 2:
        batas_atas = 1000
    else:
        print("Pilihan tidak valid.")
        return

    # Menentukan jumlah tebakan maksimum berdasarkan log2
    max_tebakan = math.ceil(math.log2(batas_atas))
    print(f"Anda memiliki maksimal {max_tebakan} tebakan.")

    # Komputer memilih angka secara acak
    angka_rahasia = random.randint(1, batas_atas)

    # Memulai permainan
    tebakan = None
    jumlah_tebakan = 0

    while jumlah_tebakan < max_tebakan:
        jumlah_tebakan += 1
        try:
            tebakan = int(input(f"Tebakan ke-{jumlah_tebakan}: "))
        except ValueError:
            print("Masukkan angka yang valid!")
            continue

        if tebakan < angka_rahasia:
            print("Terlalu kecil. Coba lagi.")
        elif tebakan > angka_rahasia:
            print("Terlalu besar. Coba lagi.")
        else:
            print(f"Selamat! Anda menebak angka {angka_rahasia} dengan benar dalam {jumlah_tebakan} tebakan.")
            return

    print(f"Maaf, Anda kehabisan tebakan. Angka rahasia adalah {angka_rahasia}.")

# Menjalankan permainan
if __name__ == "__main__":
    tebak_angka()
```

Picture 8.1 the code 8.py

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/8.py"
Permainan Tebak Angka
Pilih rentang angka:
1. 1 sampai 100
2. 1 sampai 1000
Masukkan pilihan (1/2): 1
Anda memiliki maksimal 7 tebakan.
Tebakan ke-1: 70
Terlalu kecil. Coba lagi.
Tebakan ke-2: 80
Terlalu kecil. Coba lagi.
Tebakan ke-3: 90
Terlalu besar. Coba lagi.
Tebakan ke-4: 91
Terlalu besar. Coba lagi.
Tebakan ke-5: 89
Terlalu besar. Coba lagi.
Tebakan ke-6: 85
Terlalu kecil. Coba lagi.
Tebakan ke-7: 88
Terlalu besar. Coba lagi.
Maaf, Anda kehabisan tebakan. Angka rahasia adalah 87.
```

```
PS D:\Semester 4\PrakAl_and_StrDat> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/PrakAl_and_StrDat/Module_4/Question/8.py"
Permainan Tebak Angka
Pilih rentang angka:
1. 1 sampai 100
2. 1 sampai 1000
Masukkan pilihan (1/2): 2
Anda memiliki maksimal 10 tebakan.
Tebakan ke-1: 700
Terlalu besar. Coba lagi.
Tebakan ke-2: 500
Terlalu kecil. Coba lagi.
Tebakan ke-3: 600
Terlalu besar. Coba lagi.
Tebakan ke-4: 550
Terlalu besar. Coba lagi.
Tebakan ke-5: 520
Terlalu besar. Coba lagi.
Tebakan ke-6: 510
Terlalu besar. Coba lagi.
Tebakan ke-7: 505
Selamat! Anda menebak angka 505 dengan benar dalam 7 tebakan.
```

Picture 8.2 the output.

Format penulisan laporan :

1. Kertas ukuran A4 dengan margin 2,5 cm ditiap sisi
2. Jenis font Times new roman (judul/ subjudul 12, paragraft 11)
3. Format pengumpulan dalam bentuk PDF!
4. Dibuat berdasar dengan template yang sudah diberikan dan rapi.