Soal Praktikum *Practicum Case*	
COMP6362004 Data Structures	
Teknik Informatika *Computer Science*	**O222-COMP6362004-AM01-05**
Periode Berlaku Semester Genap 2022/2023 ***Valid on*** *Even Semester Year 2022/2023*	**Revisi 00** *Revision 00*

Learning Outcomes

- LO1 – explain the concept of data structures and its usage in computer science
- LO2 – illustrate any learned data structure and its usage in application
- LO3 – apply data structures using C

Topic

- Session 5 – Binary Search Tree

Sub Topics

- Push
- Update
- Search
- Pop
- Pop All

Soal
Case

BlueBucks

BlueBucks is a brand new coffee shop in your town. The coffee shop only accepts cash as a form of payment. Currently, they are hiring a programmer to help them create an application that will help them create a system that can store the data of the customer so every transaction the customer made will be recorded as points to their membership. The criteria of the program are:

- The program consists of **4 menus**, there are:
  1. **Process Order**
  2. **View All Customer**
  3. **Remove Customer**
  4. **Exit**

```
BlueBucks
=================
1. Process Order
2. View All Customer
3. Remove Customer
4. Exit
>>
```

Figure 1. Main Menu

- If the user chooses **Process Order** (**Menu 1**), then:
  - The program will ask user for **phone number**. Validate if the inputted phone number must be **numeric** and its length **between 10 and 13**. If the inputted phone number **doesn't exist**, then:
    - The program will ask user to **input**:
      - **Name**, validate the inputted name must **start with** "**Mr.** " or "**Mrs.** " and its length must be **between 5 and 25 characters**
      - **Email**, validate the inputted email must follow the criteria below:
        - **Contains** only **1** "**@**"
        - **Ends with** "**.com**" or "**.co.id**"
        - Must be **between 10 and 20 characters**
    - After that, the program will generate a total of **10 membership points** to your account and your data **will be saved** to the **Binary Search Tree** with **phone number** as the **key**
  - If the user has **more than 25 points**, the program will ask user if he or she wants to **use their points**. Validate user can only **input** '**y'** or '**n**'.

- If the user chooses '**y**', then the program will ask user to input the **total point to redeem**. Validate the inputted point must be a **multiple of 25 and at least 25**. For **every 25 points** redeemed, the user must choose **1 drink** as their **free drink**.

➢ Then the program will ask user to **input**:

- **Drink Name**
  - Validate the inputted drink name must be **either** "**Cafe Latte**", "**Caramel Macchiato**", "**Cappuccino**", or "**Cafe Mocha**". Each drink **costs 30000**.
- **Drink Quantity**
  - **Validate** the inputted quantity must be **at least 1**.
- **Order More**
  - **Validate** the input must be **either** "**y**" or "**n**". If user chooses "**y**", the program will loop to ask user for **another drink name**. Otherwise, the program will calculate the **total price**.

➢ After that, the program will calculate the **total order** with the following formula:

$$\textbf{Total Price} = \textbf{Order}_1 + \ldots + \textbf{Order}_n$$

$$\textbf{Order}_i = 30000 * \textbf{Quantity}_i$$

➢ For every **50000 spent**, the user will **receive 3 points** (**applies multiply**). After that, **show all the order** and **total point received**.

```
Input phone number[10-13][numeric]: 0812990812312312
Input phone number[10-13][numeric]: 081299081231
Input name[5-25][Mr. |Mrs. ]: Budi
Input name[5-25][Mr. |Mrs. ]: Mr. Budi
Input email[10-20][email format]: budi@gmail
Input email[10-20][email format]: bugi@gmail.com
Insert success !


You have 0 free drink(s) left.

Input drink[Cafe Latte|Caramel Macchiato|Cappuccino|Cafe Mocha]: cafe latte
Input drink[Cafe Latte|Caramel Macchiato|Cappuccino|Cafe Mocha]: Cafe Latte
Input drink quantity[>= 1]: 0
Input drink quantity[>= 1]: 2
Do you want to order more?[y|n]: n
Order summary:
Cafe Latte        -  2x = 60000
Total: 60000
Points Obtained: 3
```

Figure 2. Non Registered User

```
Input phone number[10-13][numeric]: 081299123118
Do you want to use your points[You have 55][y|n]: y
How much[55 left]: 55
How much[55 left]: 50

You have 2 free drink(s) left.

Input drink[Cafe Latte|Caramel Macchiato|Cappuccino|Cafe Mocha]: Cafe Latte
Input drink quantity[>= 1]: 3
Do you want to order more?[y|n]: y

You have 0 free drink(s) left.

Input drink[Cafe Latte|Caramel Macchiato|Cappuccino|Cafe Mocha]: Cappuccino
Input drink quantity[>= 1]: 2
Do you want to order more?[y|n]: n
Order summary:
Cafe Latte        -  3x = 30000
Cappuccino        -  2x = 60000
Total: 90000
Points Obtained: 3
```

Figure 3. Registered User

- If the user chooses **View All Customer** (**Menu 2**), then:
  - Validate if the data is **empty** then show "**There is no data !**" message.

```
There is no data !

Press Enter to continue ...
```

Figure 4. No Data Message (View)

  - **Otherwise**, **show all customer** data using **In-Order method.**

```
-------------------------------------------------------------------------
| Name                    | Phone Number  | Email               | Points |
-------------------------------------------------------------------------
| Douglas                 | 081299123118  | douglas@gmail.com   | 55     |
-------------------------------------------------------------------------
| Ryan                    | 081288246117  | ryan@yahoo.co.id    | 28     |
-------------------------------------------------------------------------

Press Enter to continue ...
```

Figure 5. View Data in In-Order Method

- If user chooses **Remove Customer** (**Menu 3**), then:
  - Validate if the data is **empty** then show "**There is no data !**" message.

```
There is no data !

Press Enter to continue ...
```

Figure 6. No Data Message (Remove)

- **Otherwise**, ask user to input phone number. Validate if the inputted **phone number must exist** or else **show** "**Data invalid !**".

```
Input phone number: 1231231231
Data invalid !

Press Enter to continue ...
```

Figure 7. Phone Invalid

- If the inputted **phone number exists**, then **delete the data.**

```
Input phone number: 081299123118
Delete success !

Press Enter to continue ...
```

Figure 8. Delete Success

- If user chooses **Exit** (**Menu 4**), then:
  - The program will **remove all the data** from **Binary Search Tree** and **terminate** the program.

Please run the EXE file to see the sample program