


Ujian Akhir Praktikum - Semester Genap 2019/2020 <i>Practicum Final Exam - Even Semester Year 2019/2020</i>		
Matakuliah COMP6175 - Object Oriented Programming <i>Subject</i> MOBI6002 - Mobile Object Oriented Programming		
Kelas <i>Class</i>	B001 / B201 / B601 / B901 / BA01 / BA03 / BA26 / BC01 / BE01 / BG01 / BI01 / BL01 / BM01 / BT01 / BU01 / BX01 / BY01 / SB01	
	Tanggal Mulai : 18 June 2020 <i>Start Date</i>	
	Waktu Mulai : 11:20 <i>Start Time</i>	
Dosen <i>Lecturer</i>	D0208 - Drs. Agus Prahono, M.Eng.Sc. D1828 - Hendra, S.Kom., M.T. D2425 - Ir. Yasri, M.T. D3174 - Dr. Budi Yulianto, S.Kom., M.M., CBA.	Tanggal Selesai : 19 June 2020 <i>End Date</i>
	D3757 - Tegar Aryo Sulthon Musthofa, S.Kom, M.MSI D4653 - Eka Cahyadi, S.Kom., M.TI D5094 - Aswin Wibisurya, S.Kom, M.TI D5358 - Livia Ashianti, S.Kom., M.TI	Waktu Selesai : 11:20 <i>End Time</i>

PERATURAN UJIAN:

Exam Regulations:

- Mahasiswa tidak diperbolehkan berdiskusi dan/atau bekerja sama dengan peserta ujian lainnya
Student is not allowed to discuss and/or work together with other exam participants
- Mahasiswa tidak diperbolehkan menyalin jawaban dari internet
Student is not allowed to copy answer from the internet
- Asisten **BERHAK** memberi nilai 0 (**NOL**) bagi peserta ujian yang melakukan segala bentuk kecurangan
Assistant is able to give 0 (ZERO) score for exam participant who does any cheating actions
- Kumpulkan jawaban tepat pada waktunya di <https://laboratory.binus.ac.id/lab>
Submit the answer on time at <https://laboratory.binus.ac.id/lab>
- Bila Anda tidak membaca peraturan ini, maka Anda dianggap telah membaca dan menyetujuinya
If you have missed to read these regulations, so you are considered to have read and agreed on it

SOFTWARE YANG DIGUNAKAN:

Software will be used:

- Java 8
 - Eclipse Neon 3
-

FILE YANG DIKUMPULKAN:

File must be collected:

- Folder Project:
 - Bin Folder (.CLASS)
 - Src Folder (.JAVA)
-

PERHATIAN!

Attention!

- Bagi yang mengerjakan tidak sesuai dengan soal, maka akan diberikan nilai **NOL (0)**
*For those who do not work in accordance with the exam case will be marked as **ZERO (0)***
- Bagi yang mengerjakan tidak sesuai dengan software dan versi yang telah ditetapkan, maka akan tetap dikoreksi dengan software dan versi yang telah ditetapkan
For those who do not work in accordance with the software and specific version will be corrected by the predefined software and version
- Kompres semua jawaban yang akan diunggah. Pastikan format pengumpulan nama file dan ekstensi sesuai dengan format berikut: **[NIM]-[NAMA].zip**
*Compress all file that will be uploaded. Make sure the format for collecting file name and extension according to the following format: **[NIM]-[NAME].zip***

Soal*Case***Danbam****Criteria:**

1. Abstract Class

You need to design at least **three** classes, **one abstract** class, and **two concrete** classes. The abstract class consists of all **common attributes** and **behavior** that both concrete classes had. The concrete class consist of **specific** attribute and behavior that is not common between the two concrete classes.

2. Encapsulation

To **hide** the data of a class from **illegal** direct access, all the attributes of the class must be **encapsulated** and will be accessed using an **accessor** and **mutator** that may perform validation before accessing the encapsulated attribute.

3. Inheritance

All the concrete class **must inherit all** attribute and behavior from the abstract class

4. Polymorphism

If the concrete class has **a specific implementation** of the inherited behavior (method) that **differ** from the abstract class, the concrete class can **override** or **overload** the behavior from the abstract class

5. Composition

The composition concept in OOP is a strong relationship between objects that means some object must coexist together and cannot exist independently. The type of relationship with the composition concept is '**part-of**'. If any of the class contains other classes, then **composition** concept should be used to implement the feature

6. Multi-Threading

Some of the processes must use **multi-threading** to make sure the process requirement runs smoothly. The multi-threading allows the program to run with different process timeline or simply run in the background

Danbam is a new successful korean restaurant opened in Itaewon, South Korea. You are asked to create a simulation game of Danbam's daily transaction process using **Java Base Programming**. Danbam did not have any waiter as the customer is free to take any food provided in **queue of menu** made by the chef . The following criteria for the application are:

- Danbam's **initial process**:

- Each object in the program have their own **ID** in the form of **UUID** (syntax: **UUID.randomUUID()**). In the beginning of the program, danbam has **1 chef**, and **10 menus** with **the total money of 10000**

- Chef Dummy Data

Chef Id	Username	Salary	Professional Status
UUID.randomUUID()	Park Sae Royi	3000	True

- Menu Dummy Data

Menu Id	Name	Price
UUID.randomUUID()	Kimchi Jjigae 김치찌개	Random (500 - 1000)
UUID.randomUUID()	Jjinmandu 찰만두	Random (500 - 1000)
UUID.randomUUID()	Daeji Bulgogi 불고기	Random (500 - 1000)
UUID.randomUUID()	Gogigui 고기구이	Random (500 - 1000)
UUID.randomUUID()	Haejangguk 해장국	Random (500 - 1000)
UUID.randomUUID()	Sundubu Jjigae 순두부찌개	Random (500 - 1000)
UUID.randomUUID()	Saengseon Jjigae 생선찌개	Random (500 - 1000)
UUID.randomUUID()	Nakji Bokkeum 낙지볶음	Random (500 - 1000)
UUID.randomUUID()	Seolleongtang 설렁탕	Random (500 - 1000)
UUID.randomUUID()	Dolsot Bibimbap 돌솥 비빔밥	Random (500 - 1000)

- Danbam's **background process** using threading:
 - Danbam has a **queue of menu** provided in the restaurant:
 - The queue of menu is **empty** at the beginning when the program starts, so, chef has to **cook randomly from the menu** in restaurant and add it into the **queue**
 - Each chef will add menu to the **queue** for every 1 or 2 seconds depending on their professionalism
 - Danbam only has a maximum of **5 seats** in the restaurant and the customer will come until the seat full
 - Customers are free to choose **randomly** which menu in the **queue** that they want to enjoy. After the customer take a menu from the queue, **remove** that menu from the queue
 - Customers will enjoy the meal in **3 seconds** and rest for **a second** while deciding whether they want to leave the restaurant or not (the percentage of leaving the restaurant is **10%**)
 - If a customer **not leaving** the restaurant, they will choose another menu from the **queue** and keep eating
 - If a customer **leaves** the restaurant, they will **pay** for all the menu(s) in the **queue** that they have eaten (the payment will **increase** the total money of danbam and their **history orders** will be saved by the program **for each customer**) and **another customer will come** to the restaurant and enjoy the meal from the menu in the **queue**
- In the beginning, the program will **show the menus** and also the **total money** in Danbam. Remember, don't focus on designing the logo, just print "Danbam" it's enough

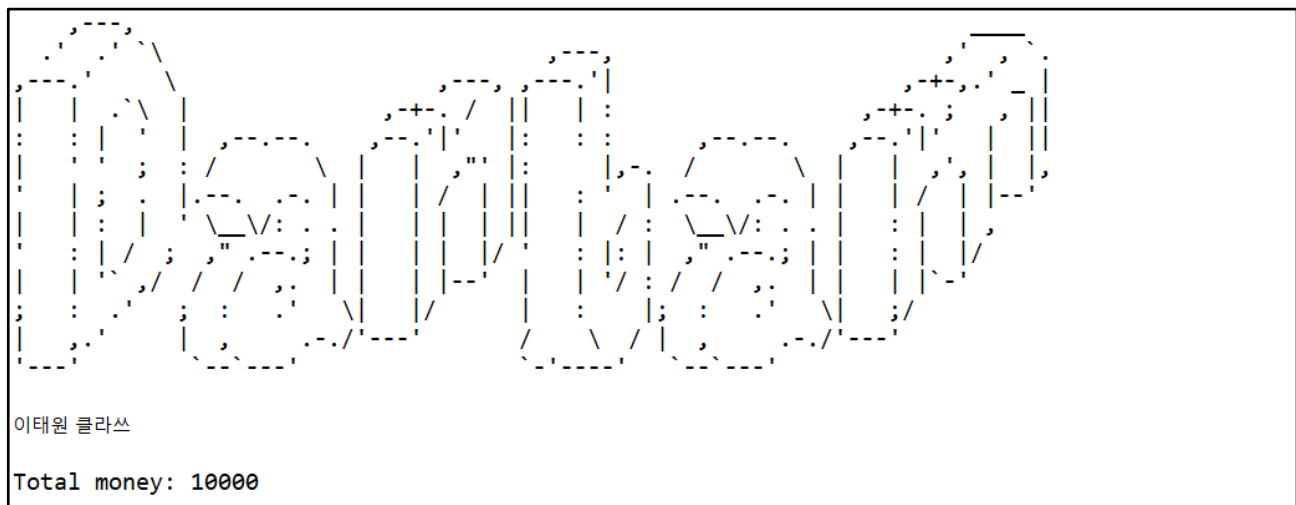


Figure 1. Main Menu Header

- The program will show **4 menus**:
 1. **Add danbam's chef**
 2. **View danbam's information**
 3. **View customers in danbam**
 4. **Quit**

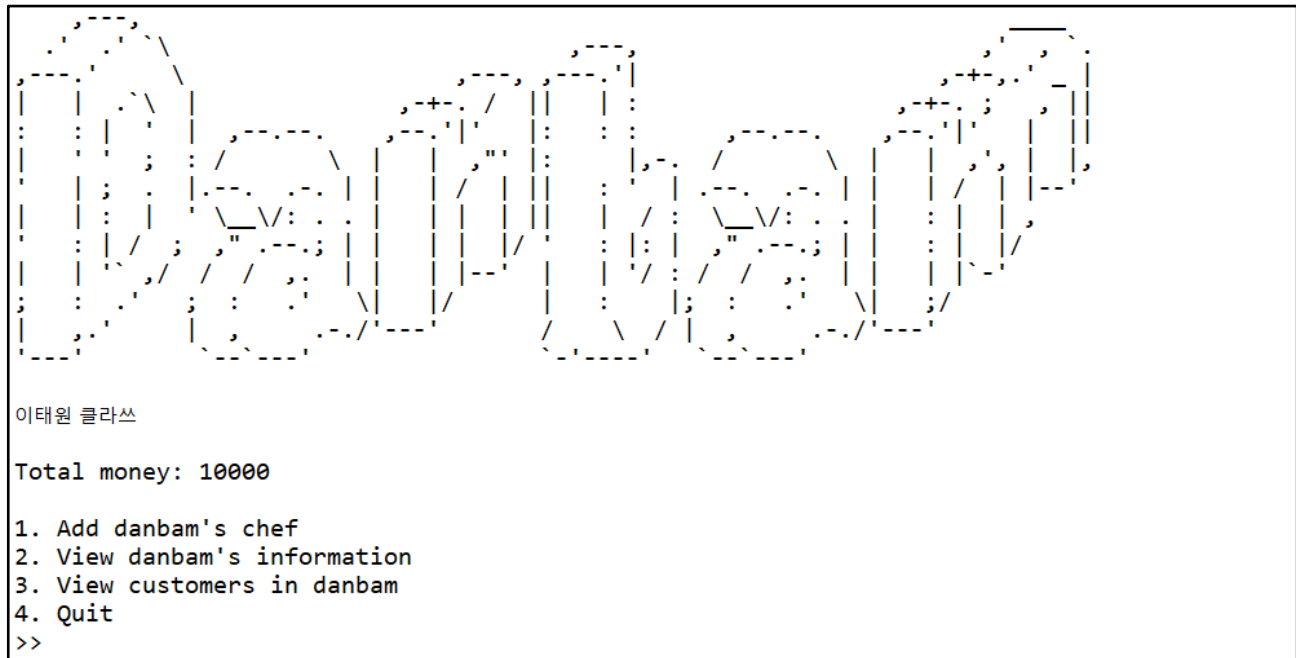


Figure 2. Main Menu

- If the user chooses **menu 1 'Add danbam's chef'**, then:
 - The program will ask the user to input the details of the chef:
 - **Chef's name**, which must consist of **exact 3 words**
 - **Chef's username**, which must be **unique**
 - **Chef's gender**, which must be **between Female and Male (case sensitive)**
 - **Chef's age**, which must be **between 17 and 40**
 - After the details of the chef has been filled:
 - **Initialize** the chef's salary with **3000**
 - **Random** whether the chef is **professional** or **not**
 - **Generate** Chef ID in the format of **UUID** (syntax: **UUID.randomUUID()**)

- Then, **add** the **new chef** into the **chef lists** in danbam
- Every chef in danbam has a **responsibility** to cook and the **right** to get salary:
 - For every **2 minutes** after the chef registration, the chef in danbam will be paid with the amount of their salary, danbam's total money will be reduced to pay for the chef
 - If the chef hired is **professional**, then the chef will be able to cook and **add the menu queue** in restaurant for every **second**. Otherwise, it is for every **2 seconds**

```
Chef's name [must contain 3 words] >> Ma
Chef's name [must contain 3 words] >> Ma Hyun
Chef's name [must contain 3 words] >> Ma Hyun Yi
Chef's username [must be unique] >> yiseo
Chef's username [must be unique] >> hyunyi
Chef's gender [Female | Male] (Case Sensitive) >> female
Chef's gender [Female | Male] (Case Sensitive) >> Female
Chef's age [17 - 40] >> 16
Chef's age [17 - 40] >> 41
Chef's age [17 - 40] >> 25

Chef has been successfully added!
```

Figure 3. Add Chef

- If the user chooses **menu 2 ‘View danbam’s information’**, then the program will display the **options menu** of which information the user want to view:

```
1. View all chefs
2. View customer with history order
3. Exit
>>
```

Figure 4. Options Menu

- If the user chooses **option 1 ‘View all chefs’** :
 - Show **all chefs in danbam**, with the details of each chef:
 - Chef Id
 - Chef’s Name
 - Chef’s Username
 - Chef’s Age
 - Chef’s Gender

```
이태원 클래스
bc6d4a72-e02d-419e-b78c-fdf9728c8794
=====
Name      : Park Sae royi
Username  : saeroyi
Age       : 33
Gender    : Male

5a21e46a-5471-4dc2-8f2e-b67ae99f6832
=====
Name      : Jo Yi Seo
Username  : yiseo
Age       : 20
Gender    : Female

ac2f2d03-b64e-4701-b551-246cc04ecb8f
=====
Name      : Ma Hyun Yi
Username  : hyunyi
Age       : 25
Gender    : Female
```

Figure 5. View All Chefs

- User must press enter to exit this view

- If the user chooses **option 2 ‘View customer with history order’** :
 - If there are none of customers with history order, then view this following message

이태원 클라쓰

There are no customers with order yet! :(

Figure 6. No Order Messages

- Otherwise **display all the customers with history order** that ever eat in danbam (This view **will be refreshed** in every second):
 - Customer Id
 - Customer's Name
 - Customer's Age
 - Customer's Gender
 - Menu details, with the details of:
 - ❖ Menu Id
 - ❖ Menu's Name
 - Total Price (obtained from the **sum of menu's price** of the **customer's history orders** while eating in danbam)

이태원 클래스

Name : Customer 2
Age : 21
Gender : Male

Menu details

```
=====
e946a9c7-2886-41b5-97e2-df317c3ed037
- Nakji Bokkeum 낙지볶음
ec638fb7-4383-4dcb-a4f8-31b76dfb24a5
- Haejangguk 해장국
```

Total Price: ₩ 1572

Name : Customer 3
Age : 20
Gender : Male

Menu details

```
=====
3fcd667a-c7ca-46de-8b7c-243f6a56b6bd
- Dolsot Bibimbap 돌솥 비빔밥
```

Total Price: ₩ 788

Name : Customer 8
Age : 21
Gender : Male

Menu details

```
=====
e946a9c7-2886-41b5-97e2-df317c3ed037
- Nakji Bokkeum 낙지볶음
```

Total Price: ₩ 936

Figure 7. View Customers with History Orders

- User must press enter to exit this view

- If the user chooses **option 3 'Exit'** :
 - Exit the following options and back to the main menu
- If the user chooses **menu 3 'View customers in danbam'**, then:
 - Show all the customers (Customer's Name) in danbam right now:

```
이태원 클래스  
  
Customer in DanBam now  
=====
```

- Customer 1
- Customer 2
- Customer 4
- Customer 5
- Customer 6

Figure 8. Customer in Danbam

- If the user chooses **menu 4 'Exit'**, then the program will be closed

If there is something you do not understand, feel free to ask your Assistant!