Kisi kisi: (200 menit UAS)

30% essay (abstract class, interface, generic, dan multi-thread)

70% case (materi uts ditambah abstract class atau interface, mungkin ada class diagram, 6-8 class)

Essay

1. What is the difference between Overloading and Overriding? Provide example code for them.
2. Interfaces are usually used to group common behaviors of class / object. Write an example by mentioning the class name and common behaviors between classes! Use Video Games context for the example.
3. What is the difference between Abstract Class and Interface? Provide example code for them.
4. Create a generic method to sort data in Ascending order.
5. Thread in java has several states. Explain those states!

Case

Mansion CareTaker

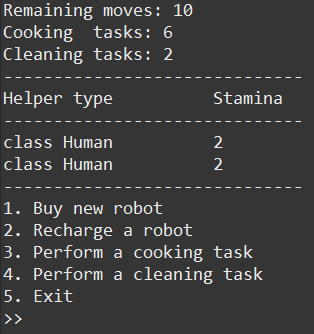
You are tasked to create a game that simulates performing tasks in a mansion using human and robot helpers in Java Programming. You must use **abstract class and interface**. It is a puzzle solving game where you have limited moves to complete the objective.

Here are the game rules:

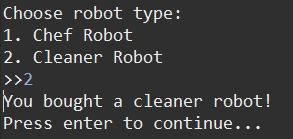
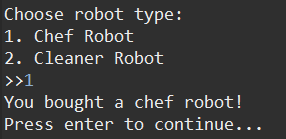
* You have **available moves of 10**. There are 2 types of tasks: **cooking and cleaning**. The game will generate a random number of at least **2 cooking and cleaning tasks**, but the **total of both tasks is 8.** You can use any random formula to fulfill this.
* You start with **2 humans** and **0 robots**. You can **buy 2 additional robots**, but you **cannot change** the number of human helpers. So choose carefully what the robot types to buy because **you cannot refund it**.
* There are 2 types of robot: **chef robot** and **cleaner robot**. Robots can only **cook** or **clean** **based on their type** while humans can do **both**.
* Both human and robot helpers have **limited stamina**. Each **human** can **perform 2 tasks (2 stamina)**, while **robots** can perform **3 tasks (3 stamina)**. When a helper has **0 stamina**, that particular helper is **no longer available to work.** Robots can be **recharged** and only restore 1 stamina.
* **Each task performed** or **recharging a robot** will **reduce the number of available moves by 1.** once available moves reach 0 and you didn’t finish all tasks, **you lose**.

Here are the details of the simulation you must create:

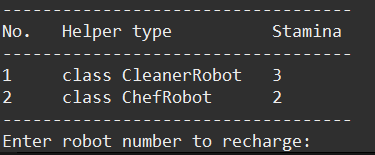
* Main menu
  + When the program starts, it will randomize the number of cooking and cleaning tasks. (See the formula above)
  + It will display: **number of available moves**, **number of cooking tasks**, **number of cleaning tasks**, and **all available human and robot helpers**.
  + Aside that, it also display additional menu choices shown in image bellow:



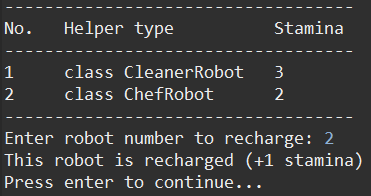
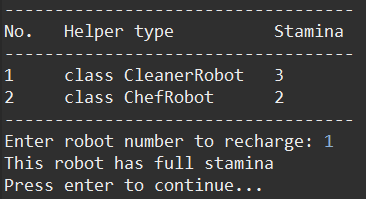
* Buy new robot
  + If you already **have 2 robots**, display the message “You cannot buy more robots!” and return to the main menu.
  + Else You will be asked to choose a robot type: **chef or cleaner.**
  + Add the robot type to your **ArrayList / Vector**.
  + Remember that once you buy a robot type, you **cannot return it!**

****

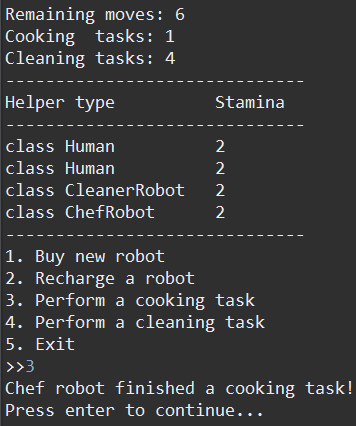
* Recharge a robot
  + If you **do not have any robots**, display the message “You do not have any robots!” and return to the main menu.
  + Else, the program will display all robots and you will be asked to **choose which robot to recharge**.



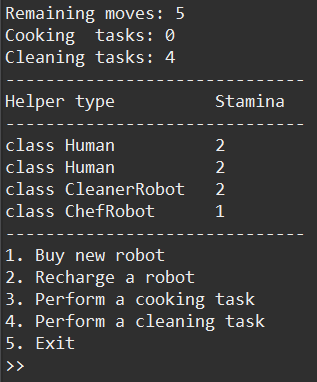
* + If you choose a robot with 3 stamina, the program will display the message “This robot has full stamina” and return to the main menu. **Otherwise**, **add 1 stamina** to the **chosen robot**, reduce the number **of available moves by 1,** and return to the main menu**.**

****

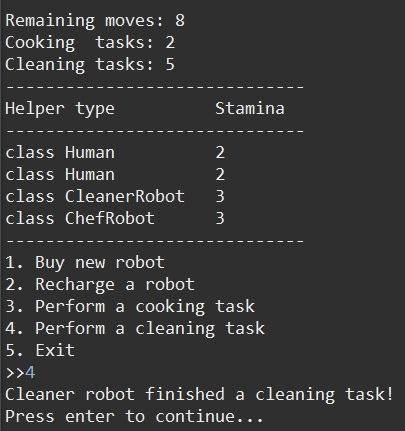
* Perform a cooking task
  + If the number of **cooking tasks is 0**, display the message “There are no cooking tasks left!”. Then return to the main menu.
  + If there are **no available human or chef robot helpers to work**, display the message “There are no available helpers to cook!”. Then return to the main menu.
  + The program will **automatically assign a human or a chef robot** to perform the task **BUT it will** **prioritize assigning chef robots** over humans.
  + Then reduce the **number of available moves**, the **number of cooking tasks**, and **stamina of the chosen human or robot** by 1.
  + For example, This is before performing a cooking task:



* + This is after performing a cooking task:



* Perform a cleaning task
  + If the number of **cleaning tasks is 0**, display the message “There are no cleaning tasks left!”. Then return to the main menu.
  + If there are **no available human or cleaner robot helpers to work**, display the message “There are no available helpers to clean!”. Then return to the main menu.
  + The program will **automatically assign a human or a cleaner robot** to perform the task **BUT it will** **prioritize assigning cleaner robots** over humans.
  + Then reduce the **number of available moves**, the **number of cleaning tasks**, and **stamina of the chosen human or robot** by 1.
  + For example, This is before performing a cleaning task:



* + This is after performing a cleaning task:

