**Electronics 2.0!**

Hi! This case will help you practicing before your laboratory final-term exam (UAP). In this case, you will use all the knowledge you have learn in this semester.

Have Fun!

When the program has been run, here’s the program flow that you need to create:

* In the beginning of program’s execution, it will **read the file** which name “electronic.txt” using the following format:

|  |
| --- |
| **[Electronic Name]#[Electronic Type]#[Electronic Price]#[Electronic Stock]** |

* If the file exists and there are contents inside, **Insert the data** and **stored it in array**. In this case, electronics storage room has a **maximum Capacity of 150 spaces**. But, to calculate spaces occupied by each item in storage room is not simply just buy adding 1. Here is the detail **on how many spaces occupied by each Item**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Electronic Type** | Small | Medium | Big |
| **Spaces** | 1 | 2 | 4 |
| **Example:**  An Electronic With type “Small” has stock 5, then it Is occupying the storage for 1 \* 5 spaces.  An Electronic With type “Medium” has stock 5, then it Is occupying the storage for 2 \* 5 spaces.  An Electronic With type “Big” has stock 5, then it Is occupying the storage for 4 \* 5 spaces. | | | |

**Table 1 Occupied Item Spaces**

* Then the program will show **four menu options**.
  1. Insert Electronic Stock
  2. View Electronic List
  3. Sell Electronic
  4. Exit

Text

Description automatically generated with low confidence

**Figure 1 Main menu**

* If user inputs **Menu 1** (Insert Electronic Stock), it will show another menu:

Graphical user interface

Description automatically generated with medium confidence

**Figure 2 Insert Option Menu**

* + **Ask** user to input **option number**, validate that user only input number between **0-2**.
  + If user **input 0**, **redirect** back to **main menu**.
  + If user **input 1** or **input 2** and **there are no more spaces** to insert data (**Occupied 150 spaces**), then show a message “**There is no spaces left**”:
  + If user **input 1** and there are enough spaces, then do:
    - * **Ask** user to input **Item Name**, validate that the name length must be **between 5-25 characters** and the name **must be unique** between all inserted electronics name (**Case-Sensitive**).
      * Next, **ask** user to input **Item Type**, validate the type must be either **Small**, **Medium,** or **Big** (**Case-Sensitive**).
      * Next, **ask** user to input **Item Price**, validate the price **must not more than the maximum Price**. Here is the **Maximum Price** for each Electronic Type:

|  |  |  |  |
| --- | --- | --- | --- |
| **Electronic Type** | Small | Medium | Big |
| **Max Price** | 200000 | 1500000 | 10000000 |

**Table 2 Maximum Prices**

* + - * Next, **ask** user to input **Item Stock**, validate the stock **must more than 0** and not make the item **occupying more than the available spaces**.
      * After all the input, Insert the data into **arrays**. And then sorted the **array** in **Ascending Order** by the Item Name (**Case-Sensitive**). After that show **message that input is success**!

Text

Description automatically generated

**Figure 3 Insert New Item Menu**

* + If user **Input 2** and there are enough spaces, then do:
    - * **View The list of electronics** that already inserted.
      * Next, **ask** user to input **Electronic Name** to add the stock of that item, validate the **Electronic Name** must be existing in the **array** (**Case-Sensitive**).
      * If user input **Electronic Name** “back” (without “”), then **redirect** back to **main menu**.
      * Else, **ask** user to input **number of stock** to be added in **the existing item**, validate stock must not make the item **occupying more than the available spaces**.

**Table

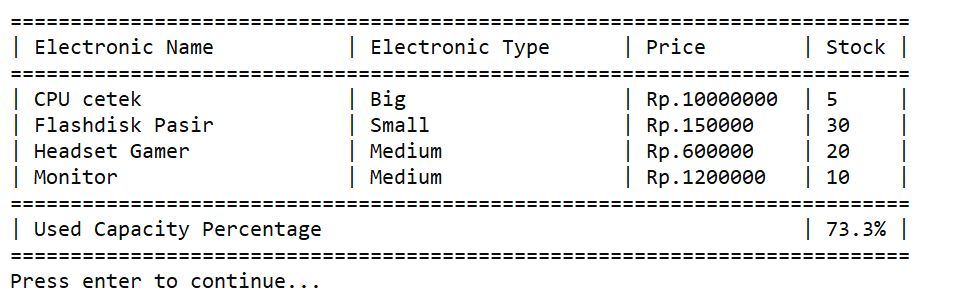
Description automatically generated with low confidence**

**Figure 4 Add Existing Item Stock Menu**

* If user input **Menu 2** (View Electronic List), do:
  + **Print** **list of items** that you have been added to the array**.** You also need to view **the percentage of used capacity** from the maximum Capacity with this formula:

**[Used Capacity Space] 🡪 total of all items occupied spaces (see calculation example at table 1)**

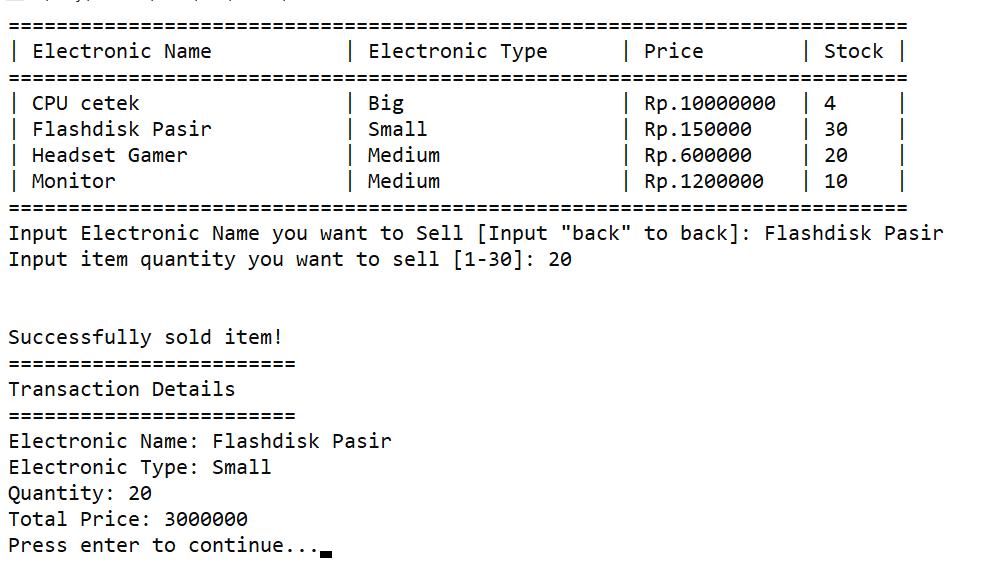
**% Used Capacity = [Used Capacity Space] / [Maximum Capacity] \* 100**



**Figure 5 list of items**

* If user input **Menu 3** (Sell Electronic):
  + **View The list of electronics** that already inserted.
  + Next, **ask** user to input **Electronic Name** to add the stock of that item, validate the **Electronic Name** must be existing in the **array** (**Case-Sensitive**).
  + If user input **Electronic Name** “back” (without “”), then **redirect** back to **main menu**.
  + Else, then do:
    - **ask** user to input **quantity** of the item to be sell, validate **quantity** must **between 1 - [Item Stock]**.
    - If user **input quantity equals to Item Stock**, then **delete** the item from the **array**.
    - Else, **update** item stock with the **subtraction of current item stock** **with the quantity.**
    - Show the Sold item details with total price that has this formula:

**Total Price = Quantity \* Item Price**



**Figure 6 Sell Item Menu**

Here are some examples of sold item in before and after:

Graphical user interface, table

Description automatically generated

**Figure 7 Sell Item 1 (Before)**

Text

Description automatically generated

**Figure 8 Sell Item 1 (After)**

**Graphical user interface, application, table

Description automatically generated**

**Figure 9 Sell Item 2 (Before)**

**Text

Description automatically generated with medium confidence**

**Figure 10 Sell Item 2 (After)**

* If user input **Menu 4** (Exit), then:
  + The program will save the electronics item **data** to the **file** name “**electronic.txt**” using the **same format as the program read before.**
  + After that the program will close.

**You can try running the program for more details!**

**If there is any question, feel free to ask your assistant via LINE.**

Good Luck 😊