BINUS UNIVERSITY
BINUS INTERNATIONAL

Assignment Cover Letter (Individual/Group* Work)

Student Information:	***Surname***	***Given Names***	***Student ID Number***
1.	Muhammad Lukman Ismail Hanafi		2301929493

Course Code : **Course Name:** Data Structures and Algorithms

Class : **L2BC**

Name of Lecturer(s) : Ir. Tri Asih Budiono, M.I.T.

Major : **Computer Science**

Title of Assignment (if any) : Final Project

Type of Assignment :

Submission Pattern

Due Date : **18 – June – 2020** **Submission Date** : **18 – June – 2020**

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating
BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality
By signing this assignment, I/we* understand, accept and consent to Binus International terms and policy on plagiarism. Herewith I/we* declare that the work contained in this assignment is my/our* own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

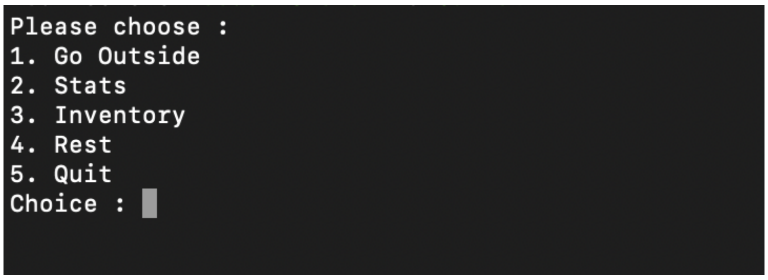
Signature of Student: *(Name of Student)*

1. [illegible] Muhammad Lukman Ismail Hanafi

Table of Contents

Collapse

Collapse is an adventure text-based RPG, you are free to do anything you want that are provided in the game, but to go to certain location you need to do some certain objective. The way you move is by input some command that are written on the program, for example if you want to open your inventory you enter the number that are written on the screen (below are the example of command)



```
Please choose :
1. Go Outside
2. Stats
3. Inventory
4. Rest
5. Quit
Choice :
```

Problem description

Inventory

First we need to know what could we used for inventory, in here I think that we could use Linked list, Queue, Stack, Vector, and Array. In the program I choose to use Array and here's why.

My first opinion would be I think having fixed size for player Inventory is better than having dynamic and flexible size, because in Array the memory is already assigned during compile time, also in most RPG. Weapon, armors and items are already predefined and here why I don't pick the other candidate.

Linked List

I think linked list is good tool since operations like insertion and deletion is fast, but In this particular game I think it's better to use Array, because in linked list you can't just add or remove a certain elements by putting the index, in linked list you have to start from the top to the elements you want, basically what am I saying is there is no random access in linked list, and since linked list is dynamic and flexible it took more memory and not only that linked list took more memory by referencing the previous and after elements.

Queue and Stack

Even though you could make Inventory by using Queue and Stack I think it's bad because they are bound to the LIFO (Last In First Out) and FIFO (First In First Out) principle, basically

if you want to use Queue and Stack for inventory its way harder because you can only insert and delete from top or bottom. Other than that Queue and Stack are also flexible and dynamic. Last opinion I think Queue and Stack is good for storing data temporarily so you could process the data one by one, but it's better to use array in this particular problem.

Vector

Vector is similar to Array, but Vector are implemented as dynamic Arrays whereas arrays can be implemented as statically or dynamically. In my opinion Vector are also similar to linked list where they are a sequential container but linked list is better because in deletion linked list could go from back and front but vector could only go from the back.

Maze

Unfortunately the program doesn't have a maze random generator but it do have BFS to search the shortest path to the goal. On why I pick using BFS is because using BFS could find the shortest route to the goal which I could reward the player if they uses the shortest path, you could also search the path by using DFS but DFS may not give you the shortest path.

Sort Inventory

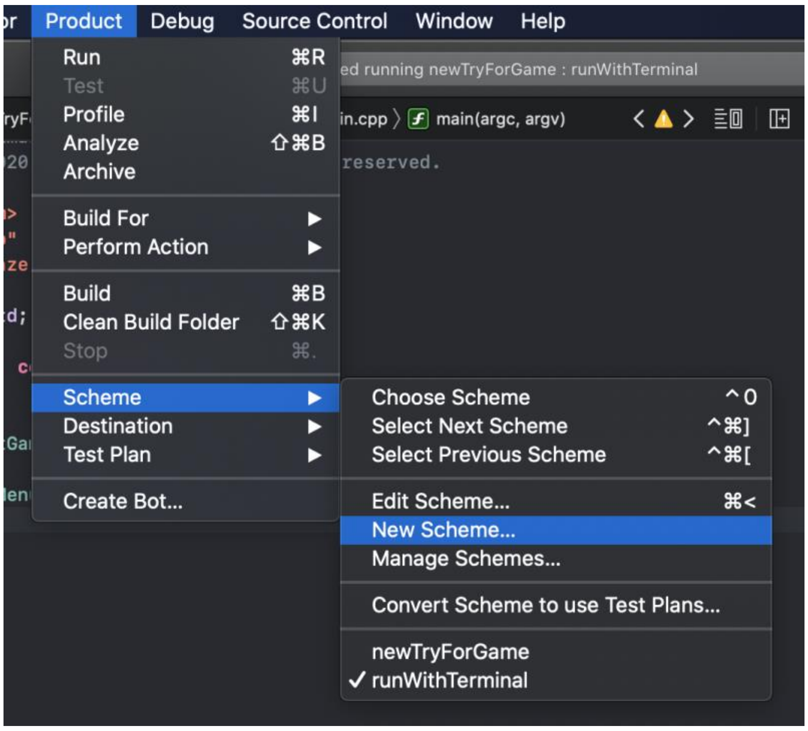
For the sorting I don't really know what sort am I using but it is close to selection sort which is the slowest from other sort but I don't have enough time to try other sort.

Program Manual

Since I'm using macOS and Xcode I could only give the proper tutorial while using said device.

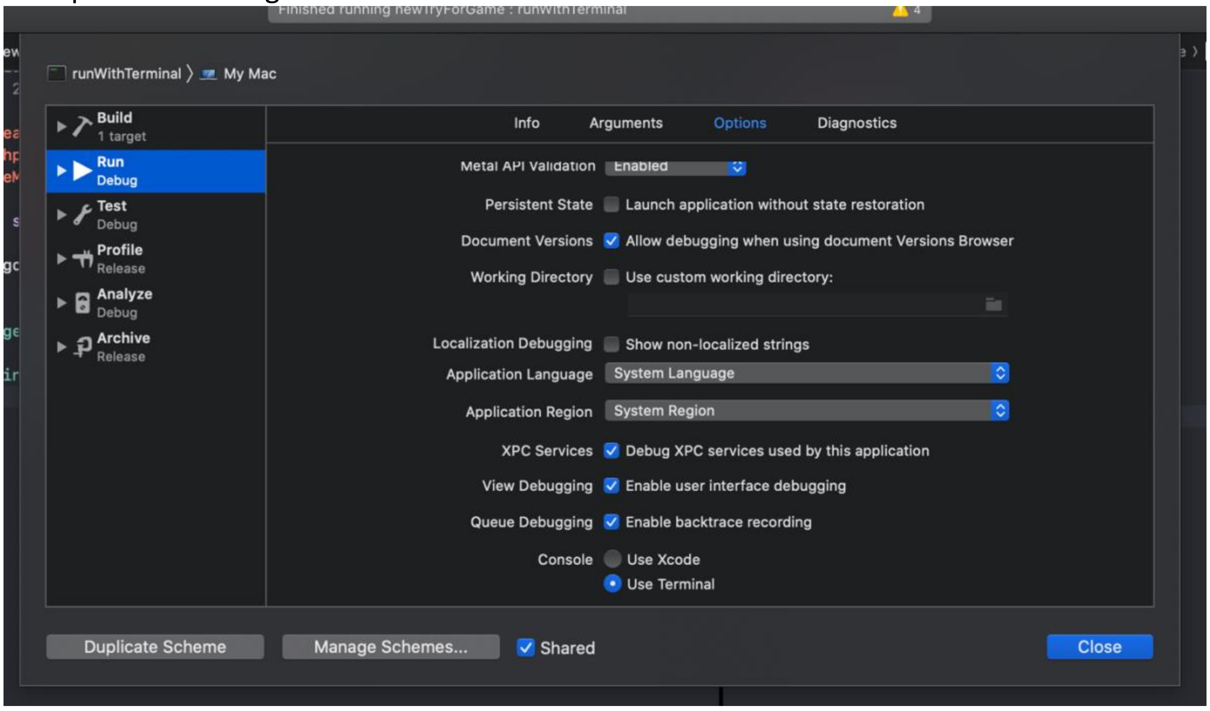
1. Make a new scheme

In Xcode you need to make a new scheme by pressing product -> scheme -> new scheme



2. Edit the scheme and change the console

After that you edit the scheme you just made, you go to the run debug and look at the option and change the console into "Use Terminal"


runWithTerminal 〉 My Mac

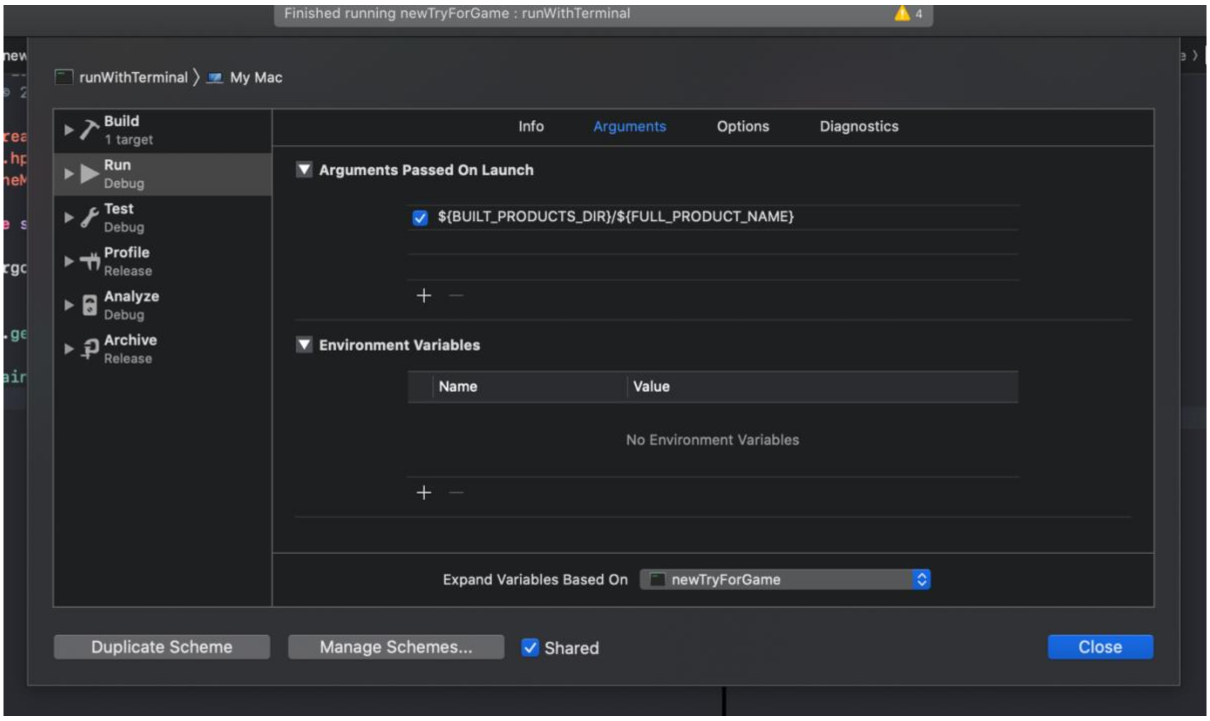
Run Debug — Options

Metal API Validation: Enabled
Persistent State: ☐ Launch application without state restoration
Document Versions: ✓ Allow debugging when using document Versions Browser
Working Directory: ☐ Use custom working directory:
Localization Debugging: ☐ Show non-localized strings
Application Language: System Language
Application Region: System Region
XPC Services: ✓ Debug XPC services used by this application
View Debugging: ✓ Enable user interface debugging
Queue Debugging: ✓ Enable backtrace recording
Console: Use Xcode [✓ Use Terminal]

✓ Shared

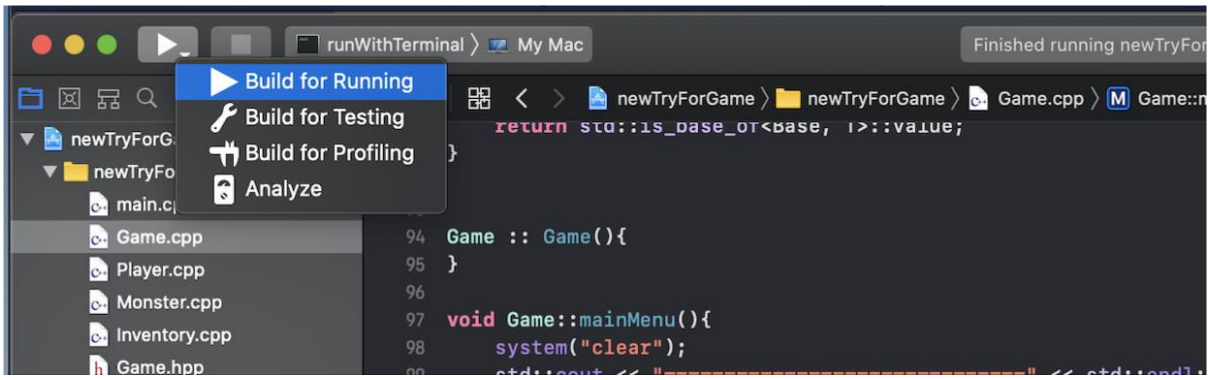
3. Change the Arguments

After changing the console into the "Use Terminal" you go to the Arguments and then type in ${BUILT_PRODUCTS_DIR}/${FULL_PRODUCT_NAME} to your arguments



```
${BUILT_PRODUCTS_DIR}/${FULL_PRODUCT_NAME}
```

4. Execute the program



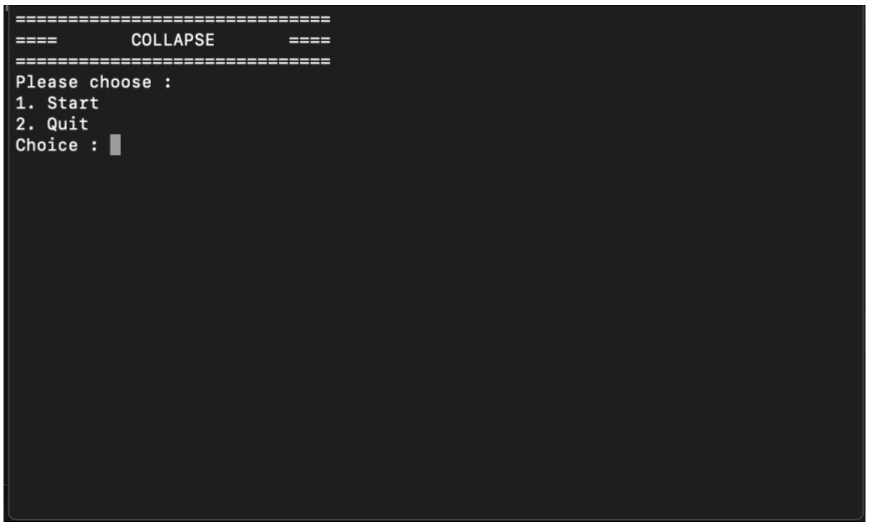
```
    return std::is_base_of<Base, T>::value;
}

Game :: Game(){
}

void Game::mainMenu(){
    system("clear");
```

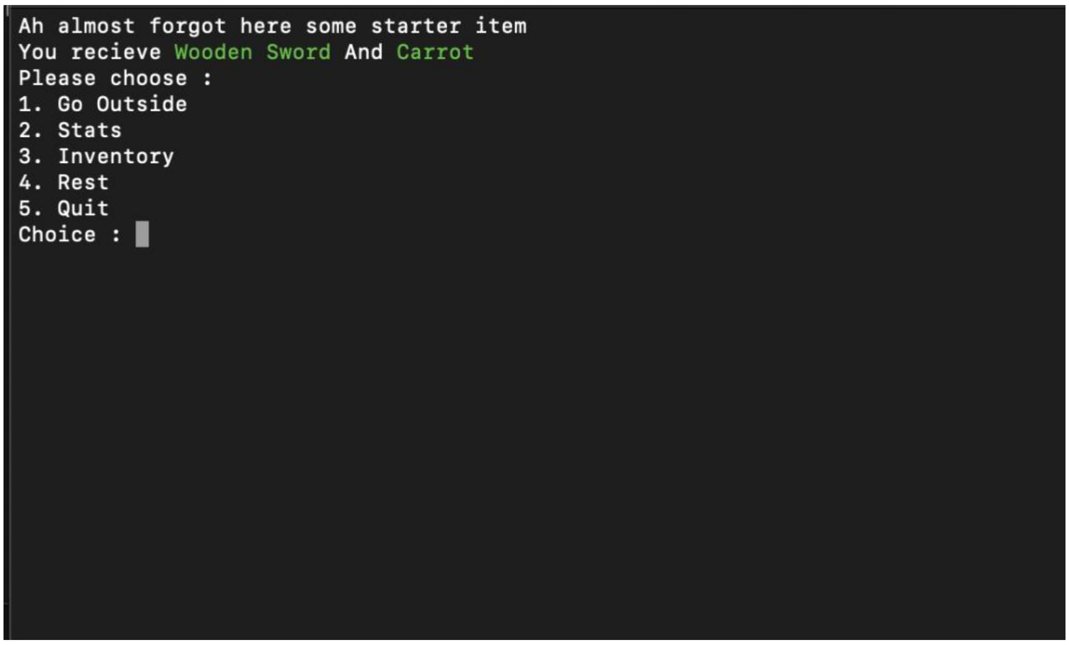
5. Result

Main Menu



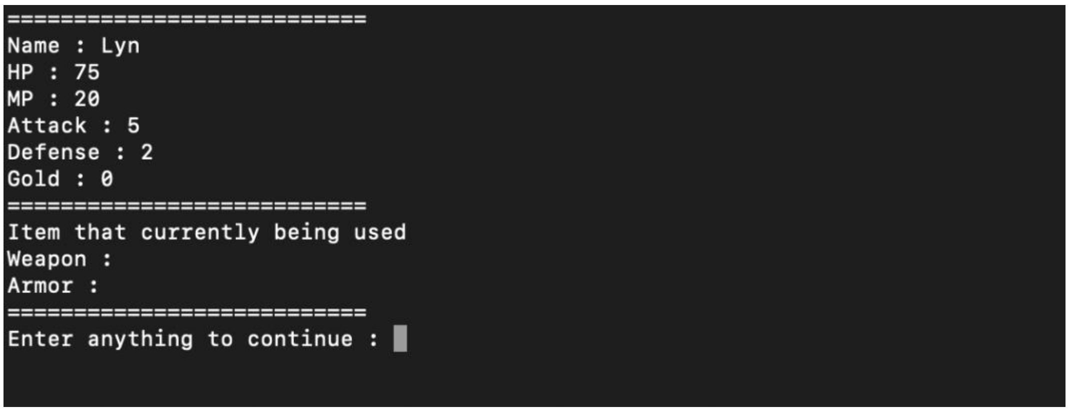
```
==============================
====       COLLAPSE       ====
==============================
Please choose :
1. Start
2. Quit
Choice : 
```

Home



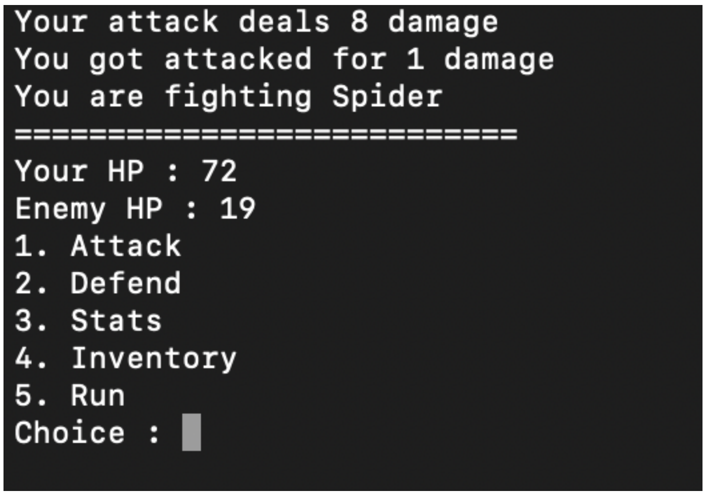
```
Ah almost forgot here some starter item
You recieve Wooden Sword And Carrot
Please choose :
1. Go Outside
2. Stats
3. Inventory
4. Rest
5. Quit
Choice :
```

Stats



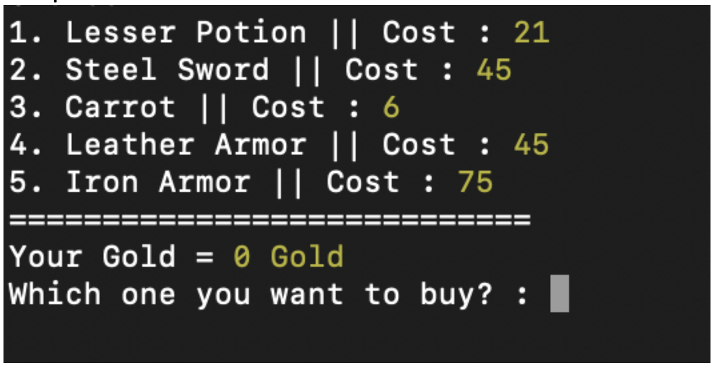
```
===========================
Name : Lyn
HP : 75
MP : 20
Attack : 5
Defense : 2
Gold : 0
===========================
Item that currently being used
Weapon :
Armor :
===========================
Enter anything to continue :
```

Battle



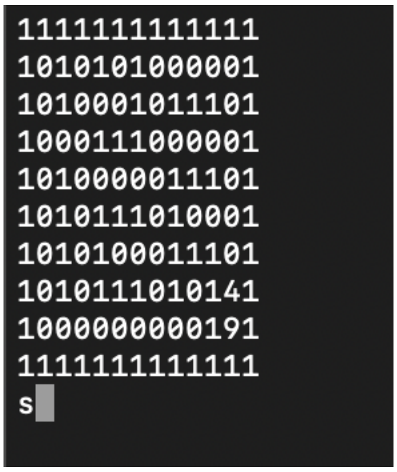
```
Your attack deals 8 damage
You got attacked for 1 damage
You are fighting Spider
===========================
Your HP : 72
Enemy HP : 19
1. Attack
2. Defend
3. Stats
4. Inventory
5. Run
Choice :
```

Shop



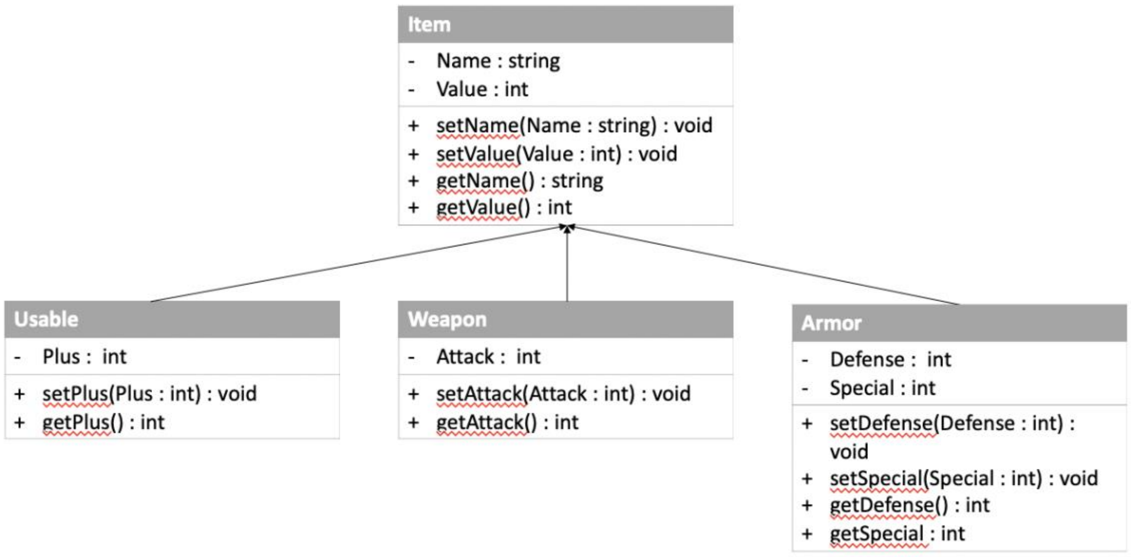
```
1. Lesser Potion || Cost : 21
2. Steel Sword || Cost : 45
3. Carrot || Cost : 6
4. Leather Armor || Cost : 45
5. Iron Armor || Cost : 75
============================
Your Gold = 0 Gold
Which one you want to buy? :
```

Maze



```
1111111111111
1010101000001
1010001011101
1000111000001
1010000011101
1010111010001
1010100011101
1010111010141
1000000000191
1111111111111
s
```

Class Diagram



This first class diagram is for the item to put inside the Inventory

Monster
- Name : String
- HP : Int
- MP : int
- MaxHP : int
- MaxMP : int
- Attack : int
- Defense : int
- Gold : int
+ setName(Name : string) : void
+ getName() : string
+ setHP(HP : int) : void
+ getHP() : int
+ setMP(MP : int) : void
+ getMP() : int
+ setAttack(Attack : int) : void
+ getAttack() : int
+ setDefense(Defense : int) : void
+ getDefense() : int
+ setGold(Gold : int) : void
+ getGold() : int
+ setDamaged(dmg: int);

Player
- Name : String
- HP : Int
- MP : int
- MaxHP : int
- MaxMP : int
- Attack : int
- Defense : int
- Gold : int
+ setName(Name : string) : void
+ getName() : string
+ setHP(HP : int) : void
+ getHP() : int
+ setMP(MP : int) : void
+ getMP() : int
+ setAttack(Attack : int) : void
+ getAttack() : int
+ setDefense(Defense : int) : void
+ getDefense() : int
+ setGold(Gold : int) : void
+ getGold() : int
+ setDamaged(dmg : int) : void
+ plusGold(Gold : int) : void
+ removeGold(Gold : int) : void
+ equipWeapon(weapon : Weapon) : void
+ unequipWeapon() : void
+ equipArmor(armor : Armor) : void
+ unequipArmor() : void
+ useItem(consumable : Usable) : void
+ plusHP(HP : int) : void
+ Weapon_Equipped : Weapon
+ Armor_Equipped : Armor

And this two other class are for the monster and player. After making this class diagram I've just realized that I should have made a new class for example "Living things" and make the player and monster inherit Living things.

Links

https://github.com/Lynceusthepotato/MRT_ORDER.git