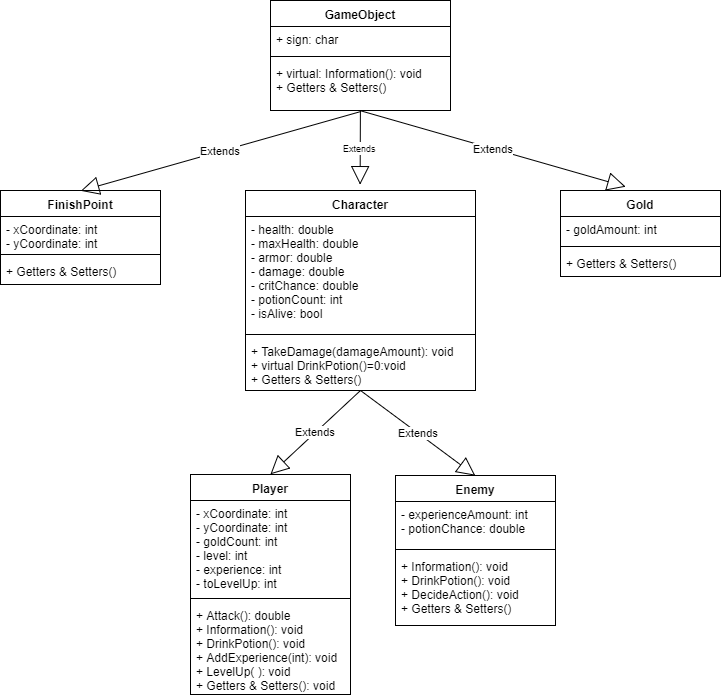
**Homework’s purpose:** This assignment aims to give you object-oriented programming experience. Usage of classes, inheritance and polymorphism is a must.

**IDE:** You must use Visual Studio 2015 in your projects. You must send your project file to the moodle after completing. Visual Studio project files can be big in size. You can use 7zip or winrar to reduce its size. Your zipped folder should be lesser than 20Mb, because moodle allows up to 20MB only. If you have a problem with uploading or Visual Studio, please contact me **before!**

**Important Notes:**

* **Name your project as nameSurname.**
* **If you’re doing this project with a group ONLY ONE OF YOU SHOULD UPLOAD THE FILE. All of your members name must be written on the project folder.**
* **There won’t be a function or algorithm restrictions, you can implement your own logic in order to create this project. But please write clean and understandable code, use appropriate variable names and functions. Example: void DrinkPotion( ), int health. USE APPROPRIATE VARIABLE NAMES THIS MAY COST YOU SOME POINTS.**
* **Use functions as much as you can.** **Write explanations whenever you can. CLEAN, UNDERSTANDABLE CODE AND EXTRA EXPLANATIONS MAY GIVE YOU SOME EXTRA POINTS.**
* **IF YOU USE SWEARWORDS OR ANYTHING INAPPROPRIATE IN YOUR PROJECTS YOUR PROJECT WILL BE GRADED AS ZERO. REMEBER, I DON’T NEED YOUR OPINIONS OR COMMENTS ON THE PROJECT AND YOU NEED GRADES.**
* **If you want to use another polymorphic approach, class structure other than the provided ones you can use your own approach on the matter. But a polymorphic structure is a must in this homework, don’t forget that.**
* **You can add or remove some components of the provided classes. The UML is given in order to create a concept. You don’t need to implement every aspect in the same way. But you must use polymorphic approach and OOP concepts in your projects.**
* **Your projects will be passed through a cheat dedection program. If your project have a similarity with someone, higher than a threshold you both get a zero. Implement your own ideas. You may cause your friend to get a zero grade too.**
* **Don’t send a project that won’t compile. I’m not going to correct your errors, you’ll get a zero. Your projects must be atleast compilable. And please don’t send me e-mails asking “Where’s my mistake in this particular code?” I won’t reply to them. It’s your job to correct your mistakes and implement your logic.**

****

**Figure 1 UML Diagram of the project**

**Question:** You’re going to implement a console RPG. The UML diagram of the classes in the game is given in the Figure 1. All functionalities are given in there but game’s life time will be explained in details.

You need to create a map that can hold: the player, enemies, a void game object (-), finish point and golds. So you must use a polymorphic structure to hold all of these objects. Character class is not instantiated anywhere on the main program, it has pure virtual functions it can be instantiated. It is only there to hold common properties of the enemy and the player class. It’d be better if you implement it like that.

**Game Mechanics:**

**Map:**

In this game there’s a map that holds every object’s position. The map printed on to the console everytime and it’ll be updated everytime player moves. You can use system(“CLS”); function ,if you’re using Windows OS, between your update operations. So it’d seem like player actually moved instead of printing the whole map again.

Game should create the map from a text file. This text file will be provided later on. Work with a temporary text file for now. In this text file the objects signs’ and position will be given. Your program should read these and create the map according to this information. You can use ‘F’ for finish point, ‘E’ for enemy, ‘-‘ for empty object, ‘G’ for gold positions. Player will always start from the bottom left corner of the map.

The map is circular, which means if the player tries to move up when it is on the top, it should be teleported to the bottom. It works for left-right, right-left, top-bottom, bottom-top ways. If there’s nothing on a specific position it’ll be displayed with ‘-‘, if the object is not in the player’s neighbourhood it’ll be displayed with ‘#’ sign (Fog of war mechanic).Look figure 3 for more details.

**Player Movement:**

Player is the protagonist in this game. Player is represented with the ‘P’ sign on the map. Player can choose to move up, down, left or right in the map. Your program must ask everytime if the player wants to move up, down, lef tor right. In that menu you can take 6 inputs. 1 for moving up, 2 for moving down, 3 for moving right, 4 for moving left, 5 to print player’s information 6 to exit from the game. Figure 2 shows the print information input’s result:

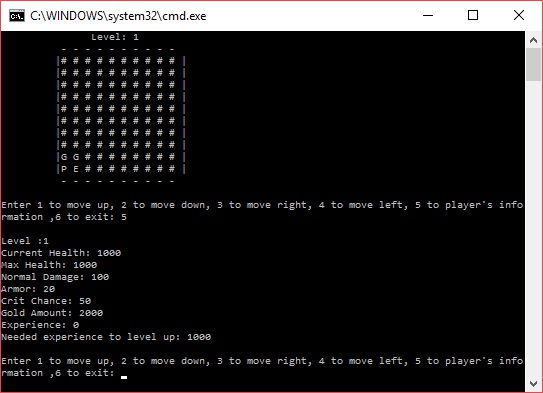


Figure 2 PrintInformation method of the player object

The player will be moved according to the input received from the user. You should check if the player can move to that position, if it can’t player must stay in its position. Example: If there’s an enemy in a way you can’t move immediately. First you must defeat the enemy and only then you can move to that position. If there’s a gold in the way player moves and collect’s it. If the finish point is on the way player moves and the game finishes after that. These will be explained later on. Look figure 3 for more details about the movement mechanic.

**Fog of War Mechanic:**

With this mechanic player can only see its immediate neighbours. All other map information will be displayed with ‘#’ sign. That means these objects are not visible to the player. You should check player’s immediate neighbours after moving the player. So you can update the immediate neighbours after every movement of the player. Ex: Let’s assume player is on the (5,5) location, (5,4), (5,6), (4,4), (4,5), (4,6), (6,4), (6,5), (6,6) will be visible to the player. Look figure 3 for more details.

**Figure 3 is a powepoint slide you can double click on it to view the slide. The slide is on a loop, after 11 movements it jumps to start the starting point. It shows the movement, map and fog of war mechanics.**

****

Figure 3 Map, Player movement and fog of war mechanics

**Collecting Gold Mechanic:**

Player can collect golds in this game. Gold objects are represented with the ‘G’ sign on the map. Golds are created from the file as explained at map mechanic. Golds values are randomized, this will be explained later on. In order to collect a gold player must move to the gold’s position. After that you should display how much gold is collected by the player. Gold amount should be transferred to the player’s currency. Lastly, the gold object that the player has collected, must be removed from the map. Collected golds are erased from the data structure. This is an important task don’t forget to implement this. Look for figure 4 for visualization.

**Figure 4 is a powepoint slide you can double click on it to view the slide and it’s also on a loop.**



Figure 4 Collecting gold mechanic

**Fighting Mechanic:**

The player can fight with the enemies. In order to fight the player should move on to the enemy. The game asks if the player wants to fight the enemy or not Y/N. If the user chooses not to fight (N) player should stay on its previous position. If the user chooses to fight (Y) game should display the enemies information first, and then ask for inputs untill player or enemy dies. It’s a turn based fight mechanic. Player can attack the enemy with the input of 1 or can drink potion with the input of 2. After each input enemy performs an action. Enemy can drink potion or attack the player.

**Attacking Mechanic:** If the player chooses to attack the enemy by entering 1, player’s damage should be sent to the TakeDamage(damage) function of the enemy object. Same thing happens fort he enemy’s attack action. Player and enemies have a critical chance. Create a random number and check if the attack is critical or not before applying damage to both sides. If an attack is critical, double the damage, if it’s not apply normal damage. Player and enemy both have armors. Apply (damage-armor) amount of damage to the both sides. If any of the sides dies return from this function with the appropriate value and continue OR end the game according to the value.

**Drinking Potion:** Both player and the enemy have a certain amount of potions at start and they can use these potions in the fight mechanic. Player can drink the potion if the user presses 2 on the fight and have atleast 1 potion. Enemy have a chance to drink a potion if it has atleast 1 potion. This chance increases proportionally with its missing health value. Ex: If the enemies health is full it’s drinking potion chance is %0, if it’s health is %20 drinking potion chance is %80. Player has 5 potion and enemies have 1 potion at start. Look for Values title for the default values.

Drinking a potion restores player/enemy health by 500. But they have a maximum health values. Their health can’t reach this value. Ex: If the player’s max health (starting health) is 1000 and it’s current health is 700 and the user chooses to drink a potion: player’s health restores up to 1000. 200 hp will not be added, 200 hp will go to waste.

**Taking a Kill:** Player and the enemy can kill their targets. If the player dies game ends. If player kills it’s enemy, a certain amount of experience point is given to the player. Experience amount is stored in the enemy object. This is a randomized value, look for the values title for more information. Player has a chance to recieve a potion after killing an enemy, this drop chance also is a randomized value. Look at figure 5 to visualize these mechanics.

**Figure 5 is a powepoint slide you can double click on it to view the slide and it’s also on a loop.**



Figure 5 Enemy attack mechanism includes drinking potion and getting experience after killing

**Leveling Up:** After killing its target player recieves a certain amount of experience. This experience can cause the player to level up. Player has a toLevelUp value (Look for values title), if player’s experience exceeds this value player levels up. You should print when this happens. After leveling up player’s stats are updated. ToLevelUp value is updated according to a formula. Ex: Let’s say the player has 800 experience and needed experience for leveling up is 1000. Player kills an enemy and recieves 500 experience. Player’s experience now setted to 300 (800+500%1000) and toLevelUp value setted to 2000 (Look for values title). Look at figure 5. Figure 5 represents the taking 500 experience after killing the enemy and displaying player’s stats after killing. Look at the updated values of the player (experience and needed experience).

**Figure 6 is a powepoint slide you can double click on it to view the slide and it’s also on a loop.**



Figure 6 Player levels up after killing the enemy

**Finishing the Game:**

The game finishes when player moves to the finishing point object. This object represented with ‘F’ sign. When player moves to its position the game finish with a appropriate output.

**Values:**

**Player:**

**Default Values:**

Health: Player starts with 1000 health

Sign: ‘P’

MaxHealth: Equals to starting health (update it after leveling up accordinly)

Armor: 20

Damage: 100

Critical Hit Chance: %8.5

Gold Amount: Player starts with 2000 gold

Experience: Player starts with 0 experience

Required experience to level up: 2000 (at level 1)

Potion Count: Player starts with 5 potions.

**After leveling up update these values:**

Level+=1

Health+=300

MaxHealth+=300

Armor+=3

CriticalChance+=0.5

Damage+=30

Required experience to level up+=1000

Experience=(Taken+Experience)%Required

**After killing an enemy player has a %30 chance to recieve a potion.**

**Enemy:**

**Default Values:**

Health: Randomized between 500-1000

Max Health: Equals to starting health

Armor:20

Experience it’ll give upon death: Randomized between 300-500

Damage: 50

Critical chance: Randomized between %4-7.5

Potion Count: Randomized between 1-2

Drink Potion Chance= Missing Healt portion. If it’s at %20 health it has a %80 chance to drink potion (if it has any left).

Enemy decides to attack if he didn’t drink potion on that turn. If it drinks it can’t attack just like the player.

**Gold:**

Gold Amount: Randomized value between 300-500

**Example: Chance of some occurence in C++:**

double critChance=7.5; //%7.5 chance

double randomNumber=rand( )%100+1;

if(randomNumber>0&&randomNumber<=critChance)

{ //That means that object landed a critical attack, do the necessary stuff in here.}