MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG242 Programming Language Concepts – Spring 2021 – Assignment 4

Date handed out: 11 June 2021, Friday
Submission due: 5 July 2021, Monday 23:55

Please Read This Page Carefully

Submission Rules

1. **You need to write a comment on the first line of your file**, stating that you read the rules specified here and the submission is your own work. **Submissions without this statement will not be graded. Example,** "/* Zekican Budin – 1234567

   I read and accept the submission rules and the important section specified in assignment file. This is my own work that is done by myself and my team-mate only */".
2. Each person is responsible seperately to write the statement on the parts they have completed.
3. As explained in the syllabus[1] provided for CNG 242, attempting any academic dishonesty[2], breaking professionalism and ethics[3] rules may result in the following:
   - You might be asked to perform an oral test to confirm your submission.
   - You may receive a "zero" grade for this submission.
   - **You may receive "zero" from all future submissions.**
   - You may receive a failing letter grade for the course, and this case might be forwarded to the discipline committee.
4. You **cannot use** someone 'else's code.
5. You **cannot hire** someone to write the code for you.
6. **You cannot share your code with someone else.**
7. You need to be ready to demonstrate your own work, answer related questions, and have short coding sessions if necessary.
8. You cannot share this worksheet with any third parties. Upon doing so, any detected action will directly be sent to the disciplinary committee. The assignment **should not be shared publicly in any manner, at any time.** The assignment cannot be disclosed or disseminated to anyone before, during, or after the submission.
9. You should read the questions fully and follow the directions listed. Only the **functions, operators, classes and/or structures with the same name will be graded.**
10. You can only get full marks, if your files fully compile, run, and generate correct output for all possible cases. **Non-general, static solutions will not be graded!**
11. **If you are added to another section as a guest student**, you will still have to submit your submission under the main section you are registered. You can check your registered section by trying to see your grades in ODTUClass. Only submit to the section that you see your grades.
12. You are not limited with any libraries. However, **do not use following** keywords; asm, auto, export, goto, mutable, register, signed, unsigned, union, volatile, wchar_t, dynamic_cast, static_cast.
13. Read the important section (Page 5) under the grading part for submission format and information.

[1] Page 3&4 (Course rules, #1,2,3)

[2] Taking unfair advantage in assessment is considered a serious offence by the university, which will take action against any student who contravenes the regulation through negligence or deliberate intent.

[3] For a comprehensive cheating definition, please refer to: https://ncc.metu.edu.tr/res/academic-code-of-ethics . When a breach of the code of ethics occurs (cheating, plagiarism, deception, etc.), the student will be added to the BLACKLIST.

Middle East Technical University
Northern Cyprus Campus

CNG242 Programming Language Concepts
Assignment 4 (C++): The Zombie Busters: Cleaning of Paradise Island from Zombie Infection
Group Project

Learning Outcomes: On successful completion of this assignment, a student will:

- Combine previously learned concepts to form a project
- Appreciate clever design and reusability of functionality

In this assignment you are going to work in groups of two.

The Game

Strategy games are fun to play mainly since they are based on the players' decision-making skills, requiring internal decision tree style thinking and good degrees of situational awareness. On the other hand, board games (even relatively simple ones) are quite popular, particularly as mobile applications. According to a set of rules, they involve counters or pieces that are moved or placed on a specifically designed surface or "board". Most of the board games would also require strategic skills, as explained above; however, it is also possible to add the element of chance to these kinds of games (e.g. dice wars[4]). Some of the well-known strategy board games are as follows: Risk, Game of Thrones: The Board Game, Puerto Rico, Diplomacy, Axis & Allies and Settlers of Catan[5].

In this assignment, you will implement a board game called "**The Zombie Busters: Cleaning of Paradise Island from Zombie Infection**". The game relies on an **imaginary** scenario, which includes two famous zombie slayers Derick Dreams, and Chichonne Mohawk. Like many other board games, we will have counters or pieces that are moved or placed on a specifically designed surface or "board", according to a set of rules. The players would undoubtedly require strategic skills. However, we are also going to include the element of chance through the use of dice. We will assume that two players Derick and Chichonne (the computer will not play the game), are positioned on random coordinates of an island in the ocean, as shown in the figure below.

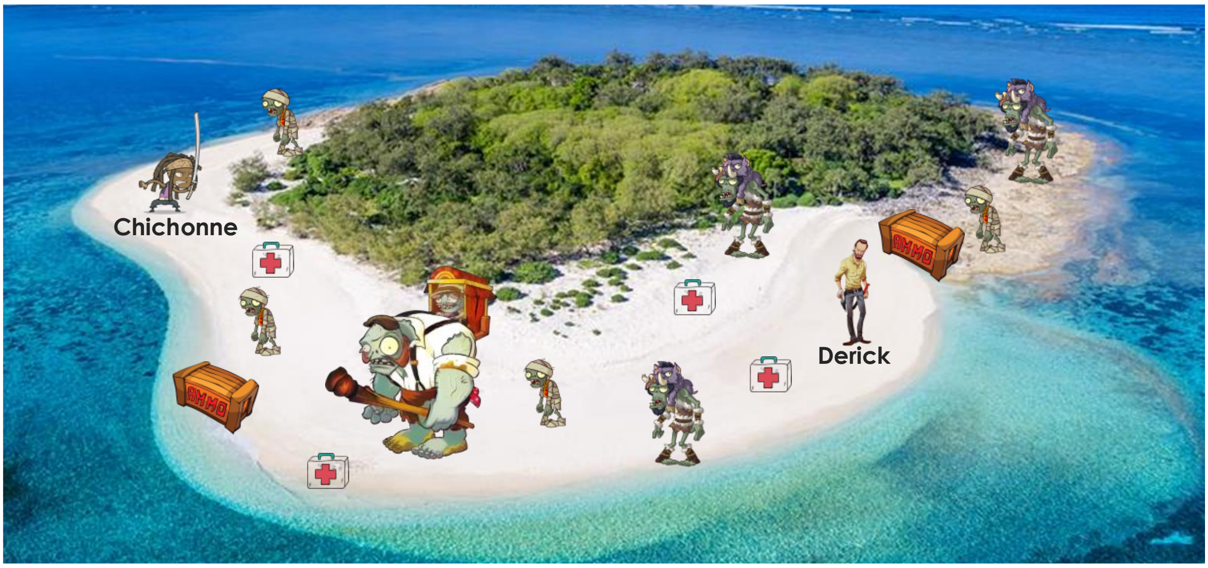
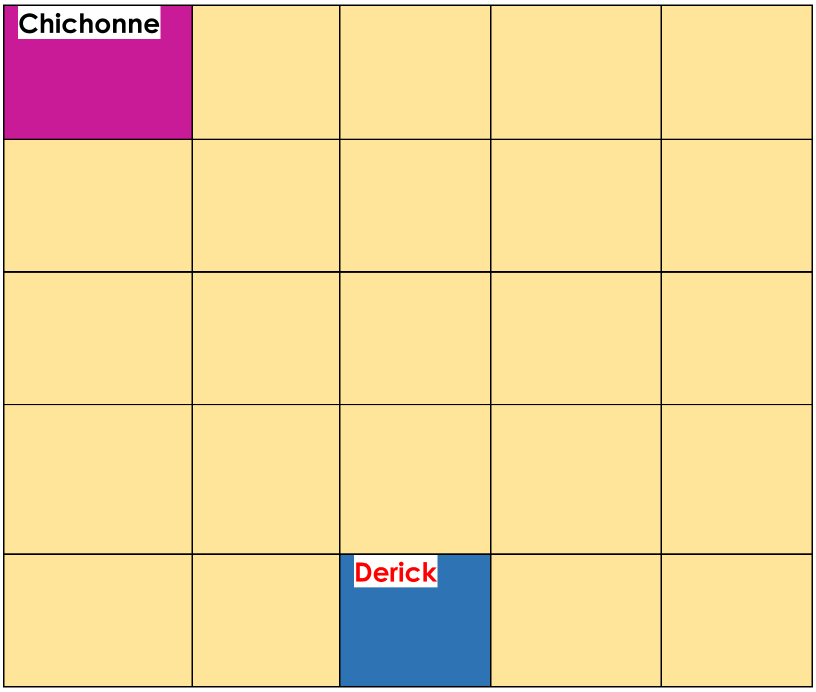


Fig.1 The imaginary paradise island

[4] https://www.gamedesign.jp/games/dicewars/

[5] https://www.wikiwand.com/en/Risk_(game)

The island is a square shape platform made of n x n elements. Derick and Chichonne are both warriors (use warrior as the parent class). The two players can choose either one of the available types(Derick or Michone type objects). Each warrior has a specific amount of life and ammunition. Derick is able to fire two shots at a time since he is a well-known gunman. Michone is able to shoot only one shot at a time; however, she has a sword that she can use (same damage as a bullet) when she is out of ammunition. Our zombie slayers will try to find all of the zombies hidden on the island and kill them. At the end of the game, if the players can kill all of the zombies and clean the island, they win. Unfortunately, there are different types of zombies hidden on this peaceful-looking island that would attack our slayers. Therefore, the life of the players decreases according to the type of the Zombie attacking.



The hidden elements can be one of the following:

- Zombies (3 types)
- Medicine kit (2 types)
- Ammunition

The size of the board for the game will be specified by the players; however, thean board is always a square shape, and the size of the board should be at least 5 x 5. For this game, the program should randomly hide $2\left\lfloor\frac{n^2}{25}\right\rfloor$zombies (at least one from each type), $3\left\lfloor\frac{n^2}{25}\right\rfloor$medicine kits, and $2\left\lfloor\frac{n^2}{25}\right\rfloor$ ammunition, resources across the grid randomly.

At the beginning of the game, the players can choose either Derick or Chichonne. Both of the characters have 100 life points. Derick has 30 ammunitions, but Chichonne has 25 at the beginning. Each shot to a zombie would reduce 5 life points from it. Chichonne has an additional weapon which is the katana she can use. At each turn, the current state of the game board should be displayed to the two players, as well as the current life points and the ammunition.

- At the beginning of each turn, the players should choose the [x] and [y] coordinates on the board.
- If there is any kind of resource at the selected coordinate, all the cells that involve the resource will become visible.
- In case there is a zombie at the selected coordinate, a one to one battle will start between the player and the Zombie. The player will start the battle by throwing the dice. Afterwards, it will be the zombies turn to attack. The Zombie will always cause the

same damage depending on its type. The battle will continue until either the Zombie or the player dies. The battle rules will be as follows:

- Derick:
  - If Derick has ammunition, he can shoot two salvos. Each shot causes 5 damage (10 in total). If there is one bullet remaining, that will cause 5 damage.
  - If Derick is out of ammunition, he can use his knife, which causes 1 damage.
- Chichonne:
  - If Michone has bullets, she can shoot only one salvo. Each shot causes 5 damage.
  - When out of ammunition Michone can use her katana which would cause 4 damage.
  - After killing 2 Zombies, the katana will level up to a mega-katana causing 6 damage.

- Dice effect (for both characters):
  - The dice effect is only valid for shooting, not for the knife of the katana.
  - One dice will be thrown for each shot
  - If the dice generates 5, or 6 this is a critical hit. Instead of 5 it will reduce the lifetime of the Zombie by 10.
  - If the dice generates 1 or 2, the wound caused is just a minor scratch. It will reduce the lifetime of the Zombie by 2.
  - The other outcomes of the dice have no effect, and the original damage of 5 life points is valid.

The objective of the game is to clean the island together; however, the players can also score and compete with each other depending on the number and sizes of the Zombies they kill. On each grid, the resource types and coordinates will be assigned automatically (randomly). The same condition holds for types and coordinates of Zombies as well. The effects of each resource can be given as follows:

Resource	**Effect**	**Size**	**Character to represent it on board**
Large Medicine Kit	Life points +20	2	*
Small Medicine Kit	Life points +10	1	+
Ammunition	Ammunition+10	1	A

The features of the three different types of Zombies can be given as follows:

Resource	**Life**	**Damage per hit**	**Size**	**Character to represent it on board**	**Score per item**
Large Zombie	12	8	3	L	100
Medium Zombie	8	4	2	M	75
Small Zombie	4	2	1	S	50

After the resources and the Zombies have been positioned, the game proceeds in a series of rounds. In each round, each player takes a turn to announce a target cell in the grid. The computer then announces the outcome of the random numbers and the actions to be followed. The grid is then updated accordingly.

Implementation

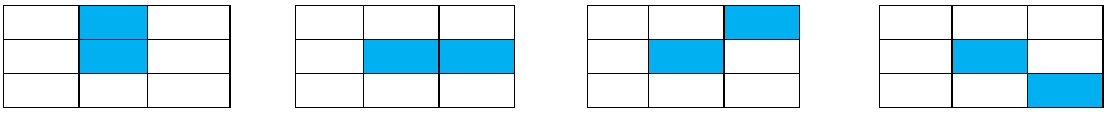
Create a base abstract class, "Entities", which will have a member variable for size and representing character. There are three sub-classes which are entities as "Warriors", "Zombies", and "Resources". These classes will be further derived for different kinds of warriors (Derick, Chichonne), Zombies (Large, Medium, Small) and resources (Small medikit, large medikit, ammunition). The Zombies killed by each player will be held in a structure (You can use any structure e.g. linked list, array, vector etc.). In each turn, the structure used is traversed to inform the players about the kills performed so far, providing information about the order of kills as well as the type of kills. An example output can be as follows:

```
Player-1(Chichonne) killed:  Large Zombie, Small Zombie, Large Zombie, Medium Zombie.
Total score: 325
```

A "Grid" class will also be created, which will be associated with the most up to date state of the island. Each player object will have access to the same grid object.

Deployment

The deploy_ function of the grid will randomly deploy resources, Zombies, and players at the beginning of the game. The resources may be placed vertically, horizontally or diagonally as in the following examples.



Inplay Turns

Each player is going to move in turns. **Before and after choosing the coordinates, the last version of the grid will be shown.** The program should not allow the players to move the same coordinates more than once.

If all the Zombies in the grid are killed, the game will end, and the program will show the scores with the details of the kills performed by each player.

Grading and Division of Tasks

Item		Mark	The programmer
Entity Class		2	Student 1
Warrior, Chichonne, Derick Classes		9	Student 1
Zombies, Resources classes		9	Student 2
Resources Subclasses		2	Student 2
Game Playing	Deployment of resources and handling the grid	9	Student 1
	The Battle Function	9	Student 2

Important (Read Here!):

1. Code modularity, pointer operations, clarity, comments will also be graded.
2. The students should work together for identifying the needs of the program.
3. All groups should be ready for demonstration after the final exam period.
4. **Please note that the fair contribution of partners is very important for us. Each student will be evaluated for individual contributions, and demonstration sessions will be conducted accordinly. In case of failure to demonstrate their own parts, the related student will be penalised following the guidelines in our sylabus.**
5. Submissions without the **submission rule** statement will not be graded.
6. If you use any library, function, class that you see online in your project, make sure to understand it fully because you might be asked to demonstrate a different implementation with the same library, function and/or class.
7. Zip your readme.txt file, .exe, .h and .cpp files (i.e. header files and/or C++ implementation files) and submit through ODTU-CLASS. Readme.txt file should include following the following:
   - Used IDE/Compiler name-version
   - Changes that you have done to project settings (if any)
   - Tested Platform (Windows 10, MacOS, Kali, Ubuntu, …)
   - Bugs that you have noticed (if any)
   - Preferred dates with time intervals for demo session (Must be after your final exam)
   - Comments about your implementation (if any)
8. Name your zip file with your student id numbers such as: 1234567_2345678.zip