

## TYPE DEFENSE

Typing games are a very well-known kind of game. They help people get quicker to type and even help them get familiar with new vocabulary or other subjects (there are even math related typing games).

### 1. Description

Your task in this assignment is to create a typing game that will take a text file as input, separate the words from one another and then display them to the player in a window so the player can try to type them

Typing a word, should make it disappear and the player should get some amount of points. The words need to be running (falling) downwards through the window in order to give the player the incentive to type faster. As an option, the speed of the words moving downwards may increase as the game goes on for a longer time (change the level).

If a word reaches the other side of the window from where it first appeared, the player loses the game. Keep track of the score, you can award points based on how far the word was from ending the game (reaching the other end of the screen). The score can be presented in any way you like.

Please feel free to play around with these ideas and create something that is fun which you could probably share with your friends.

### 2. Requirements:

This assignment can be solved with the techniques which you have learned in Modules 1 and 2. Later in this document you will see some examples. It is sufficient that you solve only one of them but if you want to do more for your practice, you may certainly do so.

- 2.1. Use a GUI-based application (doesn't need to be complicated)
- 2.2. Each word on the GUI should be a separate thread and you should be able to support AT LEAST 5 words on screen (6 threads), no reason why there couldn't be some more but not too many.
- 2.3. The user should not need to type the words in the order that they were presented
- 2.4. Each word should check the user input. In case you experience a deadlock, explain using a comment block above the methods as to what could be the reason in the code. Try to solve the situation (maybe in a copy of the method).
- 2.5. The game ends if any word reaches the other end of the window.
- 2.6. Optional: You may increase the difficulty of the game. You may raise the speed or increase the number of words gradually.

2.7. The above requirements correspond to Example 1 demonstrated below.

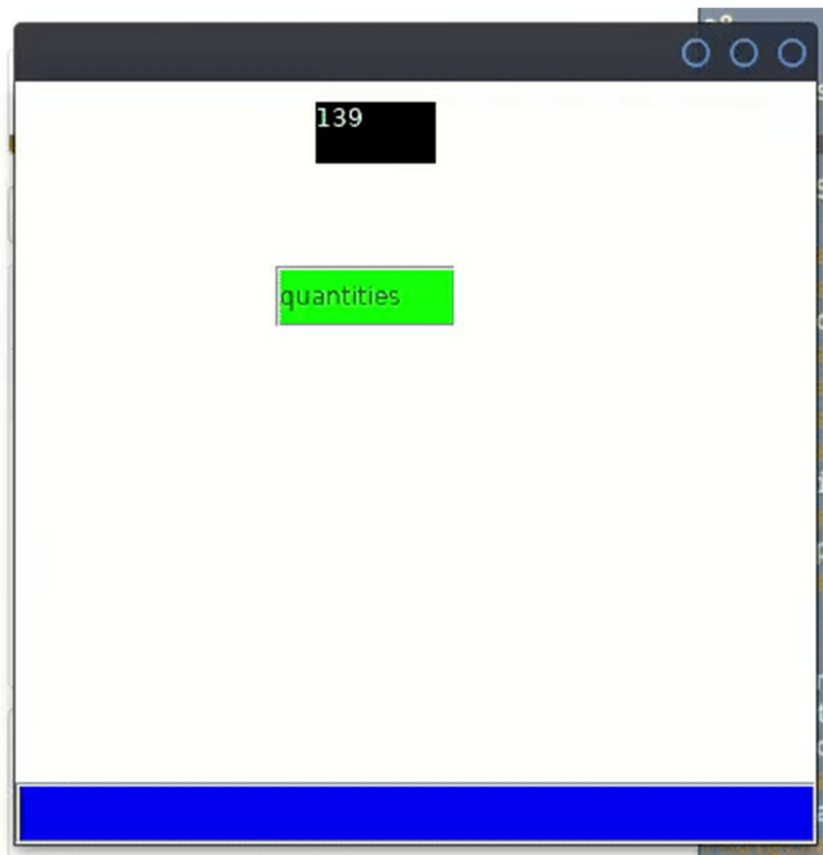
### 3. Typerman examples

If you are unfamiliar with such games and if you want to see some code solutions, the following links will help you. Remember not to copy the code and your game should **NOT** be identical to any of these; use them as reference.

**Note:** the code examples are in Java but they should be readable even if you are working with C#. The part that is quite different is handling of the GUI which in C# is in fact much easier using Visual Studio.

#### 3.1. Example 1: **Typerman**: <https://github.com/corykacal/TyperMan>

The features shown in the figure below is the minimum requirement, but you certainly do more as a bonus.








#### 3.2. Example 2: **TypeRacers**: <https://play.typeracer.com/>

In this example the user has to type everything in order, but has to race other people.

The race is on! Type the text below:

2:02

Guest (you)		55 wpm
	Guest	83 wpm
		
Guest		40 wpm
	Guest	66 wpm
		
Guest		30 wpm

This town is very peaceful. Yes, and that is why there are so many flowers. But... I have heard there are places in the world where no flowers bloom. I could never forgive myself if I did nothing to try to bring flowers to such places. That is why I shoot out spores as often as I can. It is my civic duty!

[change display format](#)

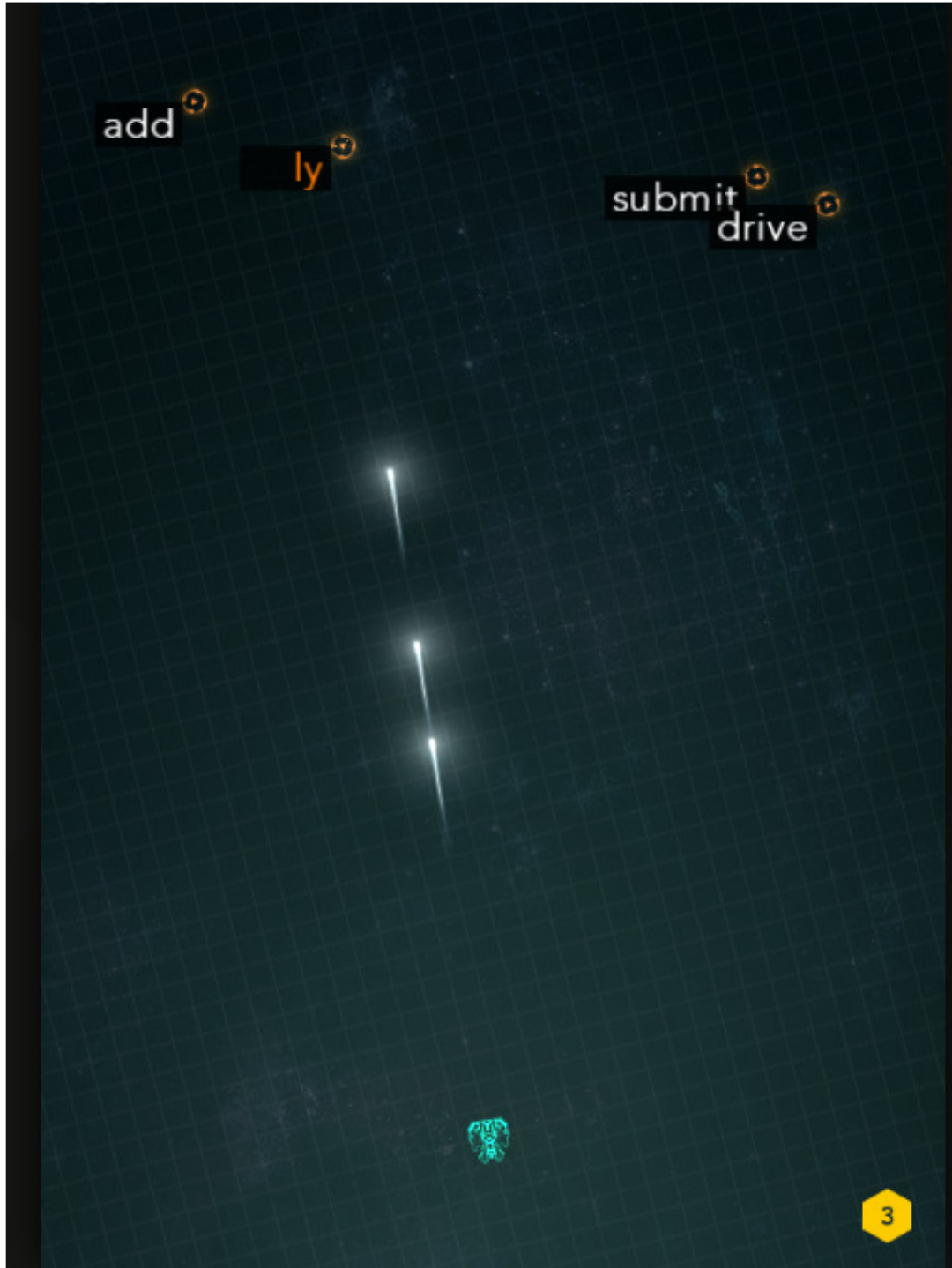
noth

?

[« main menu \(leave race\)](#)

### 3.3. Example 3: ZType: <https://zty.pe/>

The interface for this case is of course more complicated and difficult to program.



#### 4. HELP AND SUBMISSION

If you have questions or would like more clarification of any parts of this assignment, please do not hesitate to use the discussion forum in Module 2. The teachers are available to answer your questions. In turn, if you can answer others' questions or have tips and hints to share, please do that. It will be appreciated.

Submit your solution as before through Canvas. Show your assignment to your teacher during the dedicated lab-sessions.

Good Luck

Orfeas Kalipolitis and Farid Naisan