Programming assignment; undergraduate: Spring 2015

Introduction to Programming in C - ELE00002C

Exam No: Y3508038

Contents

Requirements

We were required to "*design and implement a computer game which is based on the throwing or launching of a projectile across a graphical space, avoiding obstacles and accounting for environmental effects*".

These requirements were given:

- The game should involve the throwing or launching of an object
- The path of the object must take gravity into account
- There should be at least one obstacle partially obstructing the throwing path
- There should be at least one other difficulty for the player
    - Additional "suggested" requirements:
        - Giving the game a theme
        - Adding levels of difficulty
        - Rewarding the players with a score
        - Ensuring the game play is not too frustrating

There are also some other requirements that will have to be given, due to many different platforms.

- Any device running the desktop-version of windows should be able to run it (e.g. not compatible with Linux, Mac OS X, etc.)
    - A standard keyboard with arrow keys will be needed, as will basic hardware.
    - Minimum requirements: The machine runs windows.
- The code obviously has to be written in C
    - Note: Since the compiler we're given is a C/C++ compiler, some C++ functions may be used, such as "//" comments, instead of "//*…*//)
- I will also be in need of using different libraries that we have gotten "access" to during the first term, amongst other the graphics library.

Analysis

The problem is given and the actual making of the game requires knowledge in the programming language C as well as familiarity with a compiler/debugger (Code::Blocks v13.12 used).

The program:

When the program starts, there should be a menu. After this, the actual game will consist of an object, preferably something that looks like a tanks/space-rover/similar, that the user should be able to move and place at a desired position. This object will "shoot" a projectile against a target, which the user should get three tries at. This will be happening in different atmospheres, which should be chosen along with preferred difficulty at the start. After the user has finished launching projectiles the third time, he should get a score, and the game will end.

The user:

The user will need no knowledge to play this game, as everything is explained during the game. The user will have to know how to do simple things on a computer, such as unzipping and double-clicking to be able to start the game.

Inputs:

The program should be able to run without any additional software, and is based solely on the command prompt window. Inputs will be a variation of key presses, especially(maybe);

- numbers-keys 0-9
- arrow keys
- spacebar key
- ENTER key
- Any other key may be used as well

Unexpected key-presses must be handled without the game crashing.

Outputs:

Outputs will be by text to screen, graphics, and maybe also by sound. The text will mostly appear in the command window, but text will also be printed in the graphical window to make the game easier to play/understand. The size of the command prompt window cannot change due to any code, but must be set/dragged (there are limits set in windows that handles the command prompt specifically…) to any desired size by the user. The size of the graphics window could be set to any size less than the screens resolution, but given that the user has to switch between the two to be able to play, it shouldn't be too big as you'd like to see both at once, or at least parts of both at any time. The starting positioning of the graphical window cannot be changed.

Physics integration:

As the requirements ask of the game to follow simple physics, I will have to make some statements/assumptions when it comes to units in the graphics window. E.g. I will say that 1 pixel will equal 1 meter when it comes to the actual velocity/gravity/nature-forces that will affect the projectile, but when it comes to drawing the object that throws, this will not be the case (It will be somewhat out of scale). Physics-formulas I will use/need:

Position in angular throw with forces in both x- and y direction, where θ is the angle between the throw at t=0 and the horizontal line,

- r(t) = $$\begin{matrix} x(t) = \cos(\theta)v_0 \\ y(t) = \sin(\theta)v_0 + \frac{1}{2}gt^2 \end{matrix}$$

Other values needed is surface gravity on earth, the moon and mars. (respectively: 9.81, 1.622, 3.711 m/s^2) In addition to that, basic math will be needed to calculate and draw some of the shapes that the user-controlled object is made out of, as well as in defining limits. Amongst other, pytagoras will be needed.

Specifications

The interface:

The game will consist of a welcome screen from where you can choose to either exit, play the game, or access a help-option. The help option will consist of a short walkthrough of the game. If you continue to play, you will be able to choose between low, medium or high difficulty, in addition to the atmosphere you want to play in (Either on Earth, the Moon, or on Mars) After choosing this the game starts. After the game ends, it should preferably return to the start-menu.

The game:

The game should, as stated earlier, consist of a tank-like vehicle shooting a projectile at a target three times, being able to reposition between every time. The target will be grounded. Blocking the target will be some kind of object(s) that will be placed in accordance to the choice of difficulty. Other things might be winds/solar winds or something like that. The object, as well as the winds, will change between each try. The user should receive scores after each, and a total score in the end, after which it should return to the main menu.

Mandatory requirements:

- Startup menu
- Movement of vehicle, shooting of projectile (either controlled by integers or arrow keys)
- Three tries
- Object blocking projectile path
- A target system

Optional requirements:

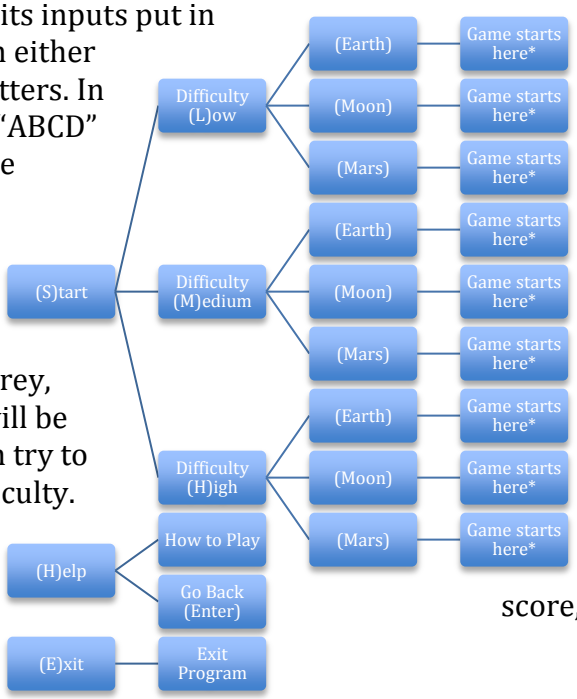
- Split startup menu with help and exit path
- Choosing of level/atmosphere with different gravity constants
- Maneuvering the vehicle is solely based on arrow keys
- Scoring system to give the user feedback on how well he did
- Return to start menu after finished game
- Winds making it challenging

Design

Start-menu:

The user will navigate his way through the menus with written commands. The commands will be simple, and will be given in each menu/submenu so that there is no confusion.

The menu is shown in this diagram with its inputs put in brackets. The inputs should be valid with either combination of lower- and upper-case letters. In choosing of difficulty, "abcd", Abcd" and "ABCD" be valid combinations for any of the three alternatives.



The game:

The game-screen will consist of the ground in accordance to the location chosen (eg. Earth – green, moon – dark grey, mars – light red), and upon it a vehicle will be placed. The obstacles will vary from each try to next, and will be different from each difficulty. There will be scores to hit on the ground. After three tries to hit the targets, the user should receive a score, and be returned to the main menu.

The vehicle will be made of

- 3 small circles as wheels
- A half circle
- A rectangle with corners in the two outermost wheels, and side through the centre of the half circle
- A "pipe"/gun-barrel

Structure:

The start-menu will be made of a series of switch-statements, setting variables for the game that will start after the location has been chosen.
The game will be made of a series of functions; drawing, erasing screen, changing position of vehicle, redrawing, setting velocity, and then finally shooting projectile. Other functions will draw obstacles, text on graphics-screen, target, etc. There will be some special functions as well, e.g. the random number seed that will have to be ran in the start of the program.

Functions:

Function(s)	Argument(s)	Return(s)	Description
main	none	0	The main function, only calls other function(s). Returns 0.
run_the_game	none	none	Functions as the main function, but within the functions-file.
random_seed	none	none	Deploys a random seed based on the system time in ms.
display_menu	none	none	Displays start-menu. Retrieves the chosen user-input.
display_help	none	none	Shows instructions on how to play. Press any key to return to the start menu.
display_difficulty	none	level_choice(int)	Displays the three different difficulties, tells the user to choose desired difficulty. Retrieves the chosen user-input.
draw_ground	ground_colour(int)	none	Draws ground.
draw_scores	none	none	Draws score-lines and labels.
draw_obstacle	level_choice(int), obstacle_coord int[array]	none	Draws obstacles.
draw_vehicle	Vehiclecoords int[array], pipecoords int[array], pipe_angle double[array]	none	Draws vehicle.
erease_vehicle	Vehiclecoords int[array], pipecoords int[array], pipe_angle double[array]	none	Draws vehicle in black.
move_vehicle	Vehiclecoords int[array], pipecoords int[array], pipe_angle double[array]	none	Checks availability to move vehicle, moves if possible.
set_angle	pipe_angle double[array]	angle (double)	Sets launch angle.
draw_projectile	pipecoords int[array], angle double[array], obstaclecoords int[array], gravity doube pointer	none	Draws projectile path.
clear_screen	ground_colour int	none	Clears screen for next throw

Variables:

Variables/arrays	Type	Range	Description
COORD[x, y]	Int [2]	0-window[x]/2, 0-window[y]	Holds x and y of center of vehicle at all times.
gravity	Double pointer	{1.622, 9.81}	Holds the gravity constant
Ground_colour	Int	Decided by level_choice	Holds the ground_color of the location chosen.
Level_choice	Int	{1,2,3}	Holds the level of choice (Low, Medium or High)
PIPE[x, y]	Int [2]	Follows COORD[x, y]	Holds the end-coordinates of the pipe.
OBSTACLE_COORD	Int [6]		Holds x,y,x,y,x,y where the 4 last are small obstacles, the first 2 are big obstacles. Needed to be like this because of the limits.

Implementation report:

Source-code:

assignment-main.c	**The main function and nothing else.**
assignment-functions.c	Every function is located here, except the main
assignment-main.h	#include's, #define's and function prototypes are located here

Changes:

Easy description	Explanation
Menu input	I considered using "ABCDE" to choose location, but figured out that an integer 1,2 or 3 was easier and chose to choose by that instead.
I had to make the position for the obstacles in arrays	Since I needed the values later on, and had two obstacles (at most) to transfer to the draw_projectile function (to set limits), I needed to change the x, x_1, y, y_1 variable to a array.
Wind	since I have the random obstacles, I decided against adding wind as I would just be causing mess in my program.
Obstacles	I had to redo the obstacle-limits as they turned out wrong the first time. Therefore the array of 5: OBSTACLE_COORD.

Testing and Verification:

Testing was done going through the game systematically and testing every input for faults, errors, etc. That meant going through all the menus, testing the limits of the projectile in every difficulty, putting every input equalt to something not asked, etc. It should now be impossible to get a run-time error, but errors may still occur due to different computers, etc.

Testing showed that f.eks. the obstacles when in difficulty 3 weren't recognised because of the different coordinates. A change was made, it was tested again and it worked.

User Manual:

To run the program, you will need a Windows computer, and be able to access Windows Explorer. Administrator rights are not needed.

Installation:

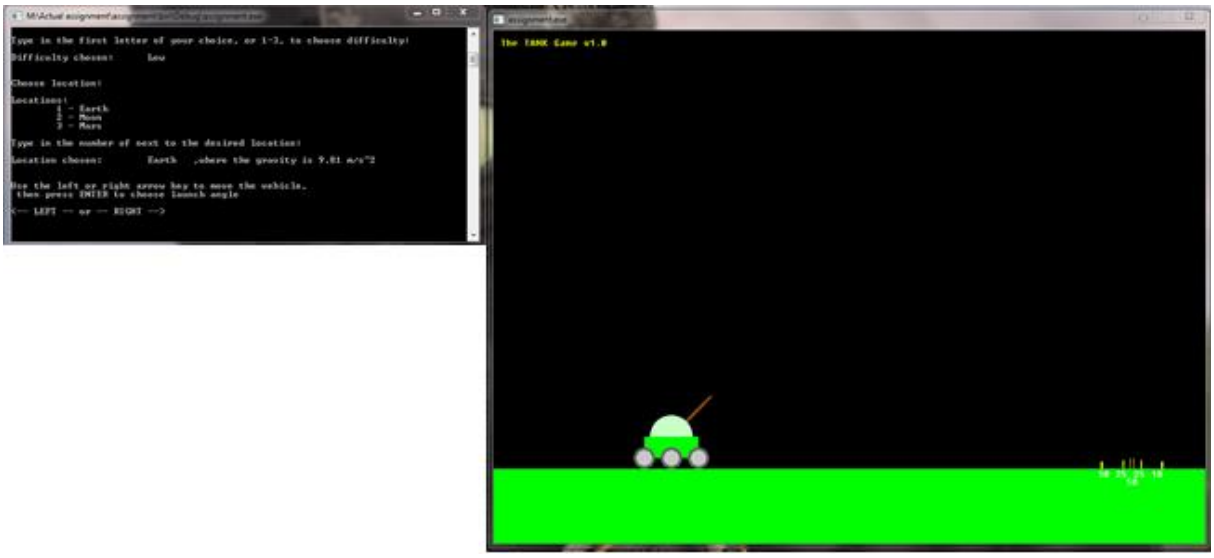
1. Download this version of the folder assignment.zip
2. Unzip the folder to any location.
3. Open the folder
4. Go to "Bin"
    a. Go to "Debug"
        i. Open the program "assignment.exe"
5. The game should now open.

System requirements:

- OS: Windows
- AS functional computer with a keyboard
    - There are no other requirements to hardware

How to play:

Use the numerical keys 1-3 to navigate the menus. After the graphical window pops up, position it to the lower right of your screen, and make the text-window(cmnd-window) active again. Example positioning:



The game is played using the arrow-keys 'LEFT' and 'RIGHT', the 'ENTER' button, and by the use of the numbers 0-9. Instructions will come by as you play if you are in doubt. Remember: Text-window needs to be in front.

Target: Hit the score markers in the ground to the left without hitting any obstacles. You have 3 tries.

Source programs:

"assignment.exe" (Program)

assignment-main.c
assignment-fucntion.c
assignment-main.h