PROJECT SPECIFICATION – "TYPERUNNER"

ELEMENTS:

André Daniel Alves Gomes – up201806224

Diana Cristina Amaral de Freitas – up201806230

DESCRIPTION:

Create a typing tutor/racer game with an animated car (video card) that moves towards a finish line (which represents a fully written sentence) and, this way, track the progress in the sentence. Furthermore, use a timer to track the performance (the speed and points) and check inputs from the keyboard to be able to play.

After the game is created. We will develop a basic menu graphic interface with 2 options : "New Game" and "Exit"; that will use the mouse's buttons to select one of the options.

To implement the movement of the mouse we will then improve the Menu by using a mouse cursor to select one of the two options by clicking on top of them. For that we will need to track the movement of the mouse.

To implement the RTC we will further improve the Menu by adding a 3rd option - "High Scores"; (while making the previous menu we will leave space for this). "High Scores" will show the best scores (ranked by words per second/ speed). With this we'll add a pre-mainMenu window where the user will insert a name and his date of birth so that we can identify his scores.

For the Serial Port we don't have definite plans because of the time schedule. However, if the game was multiplayer, it would allow users to "race" each other.

The Game sentences will be stored in a .txt file.

DEVICES:

(1) TIMER, KEYBOARD, VIDEO CARD IN GRAPHICS MODE (70%) (ALPHA STAGE)

Timer Role: Measure time interval from start to end of "run"

Timer Functionality: Interrupt

____

Keyboard Role: Serve as interface for the main part of game, the most direct input from the user: reads each key to check if sentence is being correctly typed.

Keyboard Functionality: Interrupt

____

Video Card Role: Show a progress animation for the current status of the game

Video Card Functionality: Pixel map movement horizontally

Game Screen:

A sentence is presented on screen to be typed with an animated car on top that moves along the sentence that is typed until the sentence ends OR ESC is pressed.

Post-Game Screen:

When the sentence is completed, it is presented on screen some statistics about the "run": Words per second; Success Rate; Time Elapsed

Game:

After a reverse countdown "3 2 1" the game starts:

A sentence is presented on graphic mode with a stopped car on top of it and a timer counting in real time the elapsed time.

The game's objective is to write the sentence, letter by letter without failing (errors are penalized on the final score) in the shortest time possible.

Functions to implement:

- graphstart() – Starts the graph mode with a countdown
- gamestart() – Starts the Game. Presents on screen the sentence, the car and a place where it'll appear what's been written.
    - game() – a while cycle to check keyboard's inputs. If the letter is right, the car moves a bit towards the end of it's path, the corresponding letter in the sentence turns to Green (or the background) and waits for the next input, until the end of the sentence. In case the player misses the car doesn't move, the corresponding letter turns to Red (or the background) and it becomes necessary to press BACKSPACE to return to a state where the player can continue the game. The maximum number of wrong letters is the size of the current word that is being typed, if the entire word becomes wrongly typed, the game presents a pop-up saying it is necessary to press BACKSPACE. It is possible to press ESC at anytime to exit the game.
    - results() – Shows elapsed time, percentage of completion, words per second, success rate and more.

To read make/break codes we will use 2 functions:

- A function that receives the letter and returns the make/break code

- A function that receives the sentence and returns a vector of make, break codes

(OBS: it is possible to enter 2 make codes and 2 break codes, in this succession to type, for example "ab". We will have in mind the time between the make and break codes, so that the previous is possible and it does not repeat the first letter all the time, for example "aaaab").

(2) MOUSE BUTTONS (85% CEIL.) (BETA STAGE)

Mouse Buttons Role: Select one of 2 options (RMB and LMB) for new game or Exit

Mouse Functionality: Interrupts and interpret packets to mouse presses

Functions to implement:

- MenuMouseButtons() – Shows a simple Menu in which the player can press LMB to enter the game or RMB to exit the program

(3) MOUSE MOVEMENT (85%) (GAMMA STAGE)

Mouse Movement Role: Create a cursor that moves with the mouse to select menu boxes

Mouse Movement Functionality: Interrupts and interpret packets to mouse presses and

movement

Functions to implement:

- PointerGraphic() – initializes the graphic interface in which a menu and a cursor(Moving XPM) are shown – to select menu options;

(4) (4) RTC (90%) (DELTA STAGE)

RTC Role: Record the date and time of the end of the run to keep a Leader Board

RTC Functionality: Not yet studied

Functions to implement:

- Menu() – new menu with one more option: "Show High Scores" ("Start Game" and "Exit" will be already implemented if we reach this stage) and also a screen that appears before the main menu asking for a name and date of birth.

- showScores() – shows the best scores with the same information as in results() including the sentence and the name of the player
- -results() – Add a functionality to store in a file the “runs” made with date and time

(5) X SERIAL PORT (100%) (EPSILON STAGE)

We don’t intend to implement the Serial Port because of the time schedule, but, if possible, the goal would be to allow 2 players to race with each other: with both players writing the same sentence on different computers, winner being the player that finishes first/has better words per second/better success rate . That would mean a new menu option: “Multiplayer”; and some changes in results().

(6) GAME COMPLETELY FINISHED (OMEGA STAGE)

Weekly plan:

Weeks 1 & 2 (28/11 e 5/12)

- **Video Card in Graphics Mode:** display the main graphic elements of the game: car and its path, a place holder for the user to write the sentences
- **Organize Data and Strategy:**
    - Define the strategy to show the sentences to be written
    - Sentences archive ready to use
    - Keyboard data ready to use – associate letters and symbols with its make/break codes
    - Define the strategy to implement keyboard’s functionalities

Weeks 3 & 4 (12/12 e 19/12)

- **Keyboard:** allow sentences to be written and tested
- **Video Card in Graphics Mode:** details and other functionalities that show if the letter is correct or not
- **Timer:** start implementing the timer as a mean to track the velocity (words per second) and time that took to write a sentence for example

Week 5 (26/12)

- **Timer:** implementing the timer as a mean to track the velocity (words per second) and time that took to write a sentence for example – if not yet done because of the time dispensed in keyboard functionalities
- **Video Card in Graphics Mode:** main menu implementation
- **Mouse:** main menu click/cursor options to enter or exit the game

Week 6 (2/01)

- **RTC:** Record the date and time of the end of the run to keep a Leader Board

Extra

- Sentences : http://typeracerdata.com/texts

- Break and Make Codes: http://www.quadibloc.com/comp/scan.htm