Game Prototype Assignment - April 23rd

Overview

Your goal is to have the core gameplay loop of our arcade game with minimal win/lose condition.

Submission

Each team should upload a **zip** file with a valid build inside the folder “Prototype” of the campus website before **Thursday April the 23rd at 23:59**. If more than one build is uploaded, the last one will be used. The work will be presented in the next Project class by each team (if an individual does not show up to the presentation he/she will have a “NP”). No need to create any presentation. The release 0.5 should be a **zip** containing:

- **README.md**:
  - Link to the github.com page of the project
  - Llist of team members and github accounts
  - Short description of the game
  - How to play the game, detailing the controls

- **LICENCE.md**:
  - Chose a licence that fits your project - https://choosealicense.com/

- **Game files:**
  - Executable compiled in Release with all necessary DLL files
  - A resource *folder* with all the media files (png, wav, ogg)
  - *No other file must be there!* - Be sure to remove any code and unnecessary files
  - Maximum of **20 Mb** of zipped build

- **Tasks list (pdf file):**
  - Lists of tasks delivered by each team member
  - (optional) Initial estimation time + real development time.
  - (optional) Each task should have a delivery date.

In addition, the Release folder **must** be uploaded into the Releases Github section.
The folder structure should be the following:

```
> TeamName_GameName_v.X
    > GameName
        - Assets/Resources
        - .dll files
        - GameName.exe
    - README.md
    - LICENCE.md
    - Licences (all third party libraries' licences)
```

Functionality list

This is a **minimum** funcionallity list all projects must have. Any extra funcionallities are encouraged.

Game Loop

- Initial Screen
  - Description of the project
  - Subject, degree, university and tutors
  - Team Members
  - Others: webpage, twitter account, team logo,...
- Welcome Game Screen
- Gameplay Screen: one level
- Win / Lose Screen -> back to Welcome Screen

General Gameplay

- Everything up until version 0.45 as explained in class
- Win / Lose conditions and win/lose player animations if necessary
- Camera limits: the player cannot move outside the camera
- Different background music for each screen (.ogg format)
- All necessary sound effects implemented (.wav format)
- Minumum debug functionality (F keys): god mode, display collisions, jump screens, direct win/lose,...
- Minimum UI functionality: score and lives

U.N. Squadron

- Gameplay screen must be level 2
- All enemies implemented except boss and small bosses
- One (basic) weapon type
- No health bar: player can die on single hit

Donkey Kong

- Gameplay screen must be level 4 (the one with blue flat platforms)
- No need for pick-ups, Mario can have the hammer from the beginning
- Jump not necessary

Pengo

- One level, can be hand-made (no need to be randomly generated)
- The player can destroy and shoot blocks
- If the block collides with an enemy, the block disappears and the enemy dies
- No diamonds nor wall mechanic

Pang

- First level, all the stage empty and starting with one ball
- Player movement, initial weapon, balls should split in two
- This is the minimum, I suggest you to start working on blocks and/or ladders
-

Snow bros

- First level, enemies starting on the floor at their correct positions
- Player can move, jump and shoot
- Enemies must walk around, it is not necessary that they jump
- Enemies should become a swnoball when they are hit several times
- Once an enemy starts becoming a snowball, it stops all movement
    - No need for the enemy "revival"
- The player wins when all enemies are snowballs

Acceptance Criteria

To accept the submission it must meet the following rules:

- The game is delivered on time.
- The game does not crash while testing in university's computers (a test to achieve win/lose).
- Using STL libraries is forbidden.
- Build is correct (only the necessary files, .exe compiled in Release, correct folder structure).
    - An incorrect build will have a penalization of 50% of the grade

Grading Criteria

The submission will only be accepted if they follow the rules from the *Acceptance Criteria* section.

- **Features (70%)**: All requested features are implemented and working correctly.
- **Polish (30%)**: Graphical coherence, fx implemented, fluid animations and gameplay and everything else that mimics the original game. Extra debug functionality will also be taken into account as polish.