ARCADE DOCUMENTATION

There are two Inheritances, one for the Graphical libraries, another one for the games. First you will see a short explanation of some of the functions of both inheritances, and later you will see both inheritances:

- IDisplayModule.hpp
  - virtual void init()=0; -> Function to initialize the window
  - virtual void stop()=0; -> Function to stop the window
  - virtual void initMenu() = 0; -> Function to print the Menu
  - virtual void printLevel(array_t array, unsigned int height,
  - unsigned int width) = 0; -> Function to print the maps level
  - virtual void printInfo(std::string, std::string, std::string) = 0; -> Functions
  - print the info in the Menu (the username, current game, current lib)
  - virtual void clearScreen() = 0; -> Function to clear the screen of the window
- IGame.hpp
  - virtual void update(char input) = 0; -> Function to update the position of the player on the map
  - virtual void put(unsigned int y, unsigned int x, char item) = 0; -> Function to change the array with the new position of the player

```cpp
typedef std::vector<std::vector<char>> array_t;

class IDisplayModule {

    public:
        virtual void init()=0;
        virtual void stop()=0;
        virtual const std::string &getName() const = 0;
        virtual ~IDisplayModule() = default;
        virtual bool isActive() = 0;
        virtual void refresh() = 0;
        virtual char getInput(bool) = 0;
        virtual void printLevel(array_t array, unsigned int height, unsigned int width) = 0;
        virtual std::string getUsername() = 0;
        virtual void initMenu() = 0;
        virtual bool getQuit() = 0;
        virtual void printInfo(std::string, std::string, std::string) = 0;
        virtual void clearScreen() = 0;
    protected:
        std::string _name;
        bool _quit = false;
        std::string _username;
};
```

```cpp
typedef std::vector<std::vector<char>> array_t;

class IGame {
    public:
        virtual ~IGame() = default;
        virtual void loadMap() = 0;
        virtual void update(char input) = 0;
        virtual array_t getArray() {return _array;};
        virtual void put(unsigned int y, unsigned int x, char item) = 0;
        virtual Player* getPlayer() {return _player;};
        virtual bool isGameOver() const = 0;

    protected:
        typedef struct point {
            int x;
            int y;
        }Point;
        array_t _array;
        Player *_player;
        std::set<char>SOLID_OBJECT = {VERTICAL_LINE, HORIZONTAL_LINE};
};
```