Report Workshop 2

Miguel Sanabria - 20231020153

1) **User Stories**

   1. As a customer I want to select between different types of arcade machines, so that I can buy an arcade for my taste.
   2. As a customer I want to choose the material of the arcade machine, so that my machine can be more customizable.
   3. As a customer I want to see the list of available games of the arcade machine. So that I know what games I'm going to buy with my arcade machine.
   4. As a customer I want to choose between HD and Standard definition videogames. So that I know what games I'm going to buy with my arcade machine.
   5. As a customer I want to add new games to the list of games in the arcade machines, so that I can have more games of my taste.

2) **Application Functionality**

This software was made based on the proposed solution of the workshop 1 program, it allows the customer to do the purchase of his arcade machine, let him choose between three types of materials, each of them affects the weight, power consumption and price attributes of the machine in different ways depending the material selected, the other main function is the ability to add games to the machine that is being purchased, it has more attributes than before like the information of developers, and it let the customer choose between add an HD game that is more expensive, or adding a standard definition game that is cheaper. The program uses the factory method design to create the arcade in the main class and the videogames when adding them to the arcade, this also assures that the Liskov principle of the SOLID Principles is being fulfilled.

3) Class Diagram

Main

+ valid_option(str, list)

+ add_videogame_to_machine()

+ run()

Customer

+ Name: str

+ Email: str

+ Address: str

+ __str__()

VideogameFactory

+ create_vg(int, str, str, str, str, str): Videogame

ArcadeFactory

+ create_arcade(int, str): Arcade

Videogame
<<interface>>

+ __str__()

Arcade
<<interface>>

+ add_videogame(videogame)

+ __str__()

+ get_baseprice()

+ set_baseprice(price)

+ get_weight()

+ set_weight(weight)

+ get_power()

+ set_power(power)

STD_Videogame

- Year: str

- HD_Price: int

- Category: str

- GraphicalCreator: str

- StoryCreator: str

- Name: str

+ __str__(): str

HD_Videogame

- Name: str

- StoryCreator: str

- GraphicalCreator: str

- Category: str

- HD_Price: int

- Year: str

+ __str__(): str

RacingArcade

- material: str

- dimensions: str

- weight: float

- power_con: float

- proccessors: str

- base_price: float

- videogames: []

- steering_wheel_design: str

+ add_videogame(videogame)

+ __str__(): str

+ get_baseprice()

+ set_baseprice(price)

+ get_weight()

+ set_weight(weight)

+ get_power()

+ set_power(power)

ClassicalArcade

- material: str

- videogames: []

- base_price: float

- proccessors: str

- power_con: float

- weight: float

- dimensions: str

- make_vibration: str

- sound_record_alert: str

+ add_videogame(videogame)

+ __str__(): str

+ get_baseprice()

+ set_baseprice(price)

+ get_weight()

+ set_weight(weight)

+ get_power()

+ set_power(power)

DanceRevolution

- material: str

- dimensions: str

- weight: float

- power_con: float

- videogames: []

- base_price: float

- proccessors: str

- difficulties: str

- arrow_cardinalities: str

- controls_price: float

+ add_videogame(videogame)

+ __str__(): str

+ get_baseprice()

+ set_baseprice(price)

+ get_weight()

+ set_weight(weight)

+ get_power()

+ set_power(power)

ShootingArcade

- material: str

- dimensions: str

- weight: float

- power_con: float

- proccessors: str

- base_price: float

- videogames: []

- controls_price: float

- weapons_design: str

+ add_videogame(videogame)

+ __str__(): str

+ get_baseprice()

+ set_baseprice(price)

+ get_weight()

+ set_weight(weight)

+ get_power()

+ set_power(power)