CLASH OF CLANS LITE

It's a 2D game in Python3(terminal based) , heavily inspired by Clash of Clans where the user will control the king, move it up, down , forward and backward , while destroying buildings and fighting defences on its way. Concept of OOPs is present in the code.

START

The game can be started by either king or Archer Queen .

```
Which one you want to choose King or Archer Queen? 1 for King and 2 for Archer Queen
```

The user can choose King by pressing 1 on keyboard and ArcherQueen by pressing 2.

CONTROLS

`a` moves left

`d` moves right

`s` moves down

`w` moves up

`g` applies rage spell

`c` applies heal spell

` ` space applies attack in previous direction of king and for queen in previous direction at a distance of 8 in 5x5 tiles.

`e` applies laviathan axe of king and area of effect (9x9 tiles from 16 tiles away from position of queen)

`p` `i` `n` are spawning points for **Barbarians**

`h` `u` `j` are spawning points for **Ballons**

`k` `m` `l` are spawning points for **Archers**

OOPS CONCEPT

Inheritance :

It refers to defining a new class with little or no modification to an existing class. The new class is called derived (or child) class and the one from which it inherits is called the base (or parent) class. In the code , you can see the `TownHall` `Huts` `King` are derived class of class person which gives coordinates to the object.

```
class TownHall(Person):

        def __init__(self, x, y):
            Person.__init__(self,x,y)
            self.__lives=150

            self.__destroy_flag = 0
            self.__id = "T"
```

Polymorphism :

It defines methods in the child class that have the same name as the methods in the parent class.The word polymorphism means having many forms. In programming, polymorphism means the same function name (but different signatures) being used for different types. In the code , the class `Huts` has destroy() method which destroys the huts and will destroy cannons for its derived class `Cannon` .

```
       def destroy(self,grid,arr):
            x = self.getx()
            y = self.gety()
            global BUILDING_REMAINING
            BUILDING_REMAINING = BUILDING_REMAINING-1
            for i in range(y, y+len(arr)):
                for j in range(x, x+len(arr[0])):
                    grid[i][j] = " "
```

Encapsulation :

Every component on the board is an object of a class. This instantiation encapsulates the methods and attributes of the objects. eg. Townhall , all Huts and all canons are all objects of some class.

Abstraction:

Abstraction in Python is the process of hiding the real implementation of an application from the user and emphasizing only on how to use the application.

```
    for i in range(len(Barb_arr)):
        Barb_arr[i].move(po.grid,array_cord,attack(i))
```

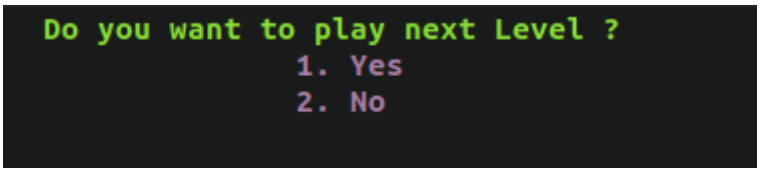
ASSUMPTIONS

- *KING HEALTH* = 150
- *BARBARIANS HEALTH* = 60
- *KING DAMAGE VALUE* = 53
- *BARBARIAN DAMAGE VALUE* = 10
- *TOWN HALL HEALTH* = 300
- *HUTS HEALTH* = 100
- *CANNON HEALTH* = 80
- *CANNON DAMAGE VALUE* = 10
- *CANNON RANGE* = 8
- *KING LEVIATHON AXE RADIUS* = 2 (AREA OF EFFECT : 5X5 SQUARE)
- *ARCHER DAMAGE* = 5

- *ARCHER HEALTH* = 30
- *ARCHER RANGE* = 5
- *BALLOON DAMAGE* = 20
- *BALLOON HEALTH* = 60
- *WIZARD TOWER RANGE* = 8
- *WIZARD TOWER HEALTH* = 80

LEVELS

There are 3 levels in the game , after winning one level the user will be asked for continuing to he next level or not. The interface will be like this .



As per the level increases the limit of troops and number of building increases.

	Level 1	**Level 2**	**Level 3**
Barbarian	7	8	9
Archers	6	7	8
Balloons	4	5	6
Cannons	2	3	4
Wizard Tower	2	3	4
Huts	6	6	6