2048 Game Documentation

2048

Windows Application

Tools used: -

◆ Python Language
- ✓ numpy
- ✓ random
- ✓ pickle
- ✓ pygame
- ✓ pyinstaller

◆ Inno Setup Compiler
- ✓ To Generate Setup file

Jyothi Prakash Muddana
(17NE1A0571)

Deevi Samhita Keertana
(17NE1A0523)

Dhulipalla Jawahar Brahma Teja
(17NE1A0526)

TIRUMALA ENGINEERING COLLEGE

Jonnalagadda,Narasaropet - 522601

DECLARATION

I the undersigned solemnly declare that the project report entitled '*2048 Game*' is based on my own work carried out during the course of B.TECH computer Science .

I assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

i.   **The work contained in the report is original and has been done by me.**

ii.   **The work has not been submitted to any other Institution for any other degree/diploma/certificate in this university or any other University of India or abroad.**

ACKNOWLEDGEMENT

I **Jyothi Prakash Muddana** student of **Tirumala Engineering College** pursuing **Computer Science and Engineering** prepared a Mini Project titled "**2048 Game**" . I sincerely express my gratitude towards **Srikanth Yadav .M** (Associate Professor Dept of CSE) . I also thankful to all the teachers who helped me to handle critical topics in the project ,I thank my friends who boasted me to complete the project successfully with out breaking it in middle

~Jyothi Prakash Muddana

Table of Contents

1.Introduction

In this project we are going to design a game called **"2048"** *this is a puzzle with numbers which helps in improving our logical skills and we can relax while playing*

To design this game, we are going to use **python language** *because of it flexibility, easy syntax, vast package resources, easy to handle*

I am going to implement **OOP** *(Object Oriented Programming) in designing this*

Packages used from python are **NumPy, random, pickle, pygame**
Every package has its own importance our main package is pygame which is used to generate (Graphical User Interface)

To generate an executable file out of the code we are going to use a package named **pyinstaller** *which is specially used to generate a specimen for code and by using that specimen it will generate our file*

To make it a perfect windows application we have to generate an executable installer and it must show out Application in control panel, this is done by using **Inno Setup Compiler**

2. Game and Rules

Here I am introducing game "2048 and its Rule"

- ➢ *Here we need a square board of (4X4) size where there is a space for 16 small squares of equal size*

- ➢ *Initially the board contains one square with some value (2,4)*

- ➢ *We can move them **(UP/DOWN/LEFT/RIGHT)***

- ➢ *If at-least one square changes its position by our moves then a new square will gets added to board at random position*

- ➢ *We can merge those small squares based on values on it by overcoming two constraint*
  - ✓ ***The Value on the Squares must be equal***
  - ✓ ***They must be next to each other (No problem if there is space in between them)***

- ➢ *If we generated a square with value 2048 then we "**WON**" the game*

- ➢ *If all the places on the board are filled and there is no chance of moving the square then "**Game Over**"*

3.Problem Identification

1. Menu Creation with following buttons
   1.1   New Game
   1.2   Continue
   1.3   Exit

2. Game Board
   2.1 Taking input from keyboard
   2.2 Moving the squares in the Board
   2.3 Merging squares based on Rules
   2.4 Declaring Game Won or Lost

3. Closing Game in middle
   3.1 Pop-up Window before closing
   3.2 Save Game (Yes/No)

4. Game Won (Popup)
   4.1 Continue Game
   4.2 Exit

5. Game Over (Popup)
   5.1 New Game
   5.2 Exit

6. Data
   6.1 Loading Data
   6.2 Retrieving data

7. Setup File for Windows GUI
   7.1 Installer
   7.2 Uninstaller

4.System Requirements

System requirements are the required specifications a device must have in order to use certain **hardware** or **Software**

- ## *Hardware*
  Computer hardware refers to the physical parts of a computer and related devices. Internal hardware devices include motherboards, hard drives, and RAM. External hardware devices include monitors, keyboards, mice, printers, and scanners.

- ## *Software*
  Computer software is a general term that describes computer programs. Related terms such as software programs, applications, scripts, and instruction set all fall under the category of computer software.

4.1 Hardware requirements

- ✓ Processor (minimum 2GH or More)
- ✓ Hard disk (minimum 36GB or More)
- ✓ Ram (minimum 4GB or More)
- ✓ 5 GB free diskspace
- ✓ All other Basic requirements like keyboard, Mouse etc... for a normal PC requires

4.2 Software requirements

- ✓ Windows operating system (7 or 10)
- ✓ Inno setup Compiler
- ✓ Python 3.6 or above version
  - ❖ Required packages
    - o NumPy
    - o random
    - o pickle
    - o pygame
    - o pyinstaller

5.Theoretical background

5.1 Python

◆ **What is python, its importance and usage**

Python is a popular programming language. It was Developed by **Guido van Rossum**, and released in 1991.

Now a days we can see its foot prints in web development (server side), software development, Data Science, Artificial Intelligence, etc···

Most import features of python
- Readable and Maintainable Code
- Multiple Programming Paradigms
- Compatible with major platforms and systems
- Robust standard library
- Many open source frameworks and tools
- Simplify complex software development
- Adopt text driven development

◆ **How to Install python**

We can Install python on many Operating Systems such as Windows, linux/Unix, Mac OS and others

To download python installers, use below link
https://www.python.org/downloads/

This is official python website and it will detect he operating system and based on that it will recommend you to download python

You can install python in u computer just like an application but make sure to give a checkmark to add paths to the system while installing itself will help you to install other required package using pip in command prompt

◆ **Packages that are required for this project**
- NumPy
- pickle
- random
- pygamer
- pyinstaller

5

5.1.1 NumPy

NumPy is a library from python programming language, adding support for large, Multidimensional arrays and matrices, along with the large collection of high-level mathematical function to operate on these arrays

It is a project by a Community release in 1995 as Numeric later in 2006 rename it as NumPy and written in python, c It contains all precompiled function makes its speeder than other basic function of python

To install this library, use the following command in CMD
> **>pip install numpy**

I am going to use this module to use 2D arrays as matrix to create a virtual Game-board

5.1.2 random

Random is a library from python, which is used to generate random numbers, it is most used package for randomizing thing, it is used to generate random decimal values for 0 to 1, Integers of specific range, shuffling list, choosing a random value out of list etc···

To install this library, use the following command in CMD
> **>pip install random2**

I am going to use this to choose random position on gameboard to insert square and to randomize the value on square

5.1.3 pickle

Pickle module in python is used for **serializing** and **de-serializing** a python object structure. Any object in python can be pickled so that it can be saved on disk

We can't store object in disk so it will convert into another format in which we can store the object in disk
- o serializing: - In this object is converted something like into text format
- o DE-serializing: - form text format to object again

No need of installation to use this module by default it is available in python

I am going to use a text file as my database by using pickle module

6

5.1.4   *pygame*

It is a cross-platform set of python modules designed for writing video games . It includes computer graphics and sound libraries designed to be used with the python programming language

Pygame is highly portable and runs on nearly every platform and operating system

Pygame is free. Released under the LGPL licence, You can create open source, freeware , share ware ,and commercial games with it.

To install this library, use the following command in CMD
   >pip install pygame
I am going to use this module to give some graphic to the game like as follow
1. Creating a window
2. Menu buttons
3. Motion animation for Squares
4. Popups
5. For taking input(up,down,left,right) arrows etc···

5.1.4 *pyInstaller*

It freezes (packages) python applications into stand-alone executable under windows,GNU/LINUX,Mac Os X, ···

The main goal of PyInstaller is to be   **compatible with 3rd-party packages out-of-the-box**. This means that, with PyInstaller, all the required tricks to make external packages work are already   **integrated within PyInstaller itself**   so that there is no user intervention required. You'll never be required to look for tricks in wikis and apply custom modification to your files or your setup scripts. As an example,   **libraries like PyQt, Django or matplotlib are fully supported**, without having to handle plugins or external data files manually.

To install this library, use the following command in CMD
   >pip install pyinstaller

I am going to use this module to generated and exe file out of python code

5.2   Inno Setup compiler

Inno Setup is a *free* installer for Windows programs by Jordan Russell and Martijn Laan. First introduced in 1997, Inno Setup today rivals and even surpasses many commercial installers in feature set and stability.

Key features:

- Support for every Windows release since 2006, including: Windows 10, Windows 10 on ARM, Windows Server 2019, Windows Server 2016, Windows 8.1, Windows 8, Windows Server 2012, Windows 7, Windows Server 2008 R2, Windows Server 2008, and Windows Vista. (No service packs are required.)

- Extensive support for installation of **64-bit applications** on the 64-bit editions of Windows. The x64, ARM64 and Itanium architectures are all supported.

- Extensive support for both administrative and non administrative installations.

- Supports creation of a **single EXE** to install your program for easy online distribution. Disk spanning is also supported.

- Standard Windows wizard interface.

- **Customize setup types**, e.g. Full, Minimal, Custom.

- Complete **uninstall** capabilities.

- Installation of files:
  Includes integrated support for "deflate", bzip2, and **7-Zip LZMA/LZMA2 file compression**. The installer has the ability to compare file version info, replace in-use files, use shared file counting, register DLL/OCX's and type libraries, and install fonts.

- Creation of shortcuts anywhere, including in the Start Menu and on the desktop.

- Creation of registry and .INI entries.

- Running other programs before, during or after install.

- Support for **multilingual** installs, including right-to-left language support.

- Support for passworded and encrypted installs.

- Support for **digitally signed** installs and uninstalls, including dual signing (SHA1 & SHA256).

Use below link to download and install the Inno setup compiler   (ctrl+click)

https://jrsoftware.org/download.php/is.exe

I am using this **(Inno SetUp Compiler)** to generate a **Setup** file for the generated exe file to create a process for **(Installing/Uninstalling)**

6.   *Code Documentation*

For a programmer reliable **documentation** is always a must. The presence of **documentation** helps keep track of all aspects of an application and it improves on the quality of a software product. Its main focuses are development, maintenance and knowledge transfer to other developers.

Successful documentation will make information easily accessible, provide a limited number of user entry points, help new users learn quickly, simplify the product and help cut support costs. Documentation is usually focused on the following components that make up an application: server environments, business rules, databases/files, troubleshooting, application installation and code deployment.

The code documentation is the backbone of every application. Code documentation can be split in multiple parts. The first one, the most helpful for programmers are the comment blocks. These will be found through every file explaining classes, methods, parameters, possible errors. Then comes the specific file documentations. These are usually generated through a third party script which will parse a file and, based on the comment blocks, will create an explicit PDF. Afterwards there should be information regarding the code repository, where the file updates are found, and where they need to be moved. In addition, there should be step-by-step instructions on how to create an application package or a build to be deployed.

In Python to document a code by default we have a module called pydoc

6.1   Pydoc

This is in build available in python

Pydoc is the standard documentation module for the programming language Python. **Pydoc** is used to extract documentation from the source code itself. ... More comprehensive documentation is generated from external restructured-text documents using the Sphinx documentation system.

Out of our comments and import information in our it generates its documentation in text format if need we can get the information in HTML also

6.2   How to document code

It is very easy by using pydoc just use   ("""   """ or ''' ''') use either of these two with in the function or class after defining it and write its importance point like logic, about data, interconnections etc ...

Eg :-

    def add(a,b):
        '''   This function is used to add two numbers
            And returns its sum
        '''
        return a+b

<mark>This part comes under documentation</mark>

6.3   Local Host (Document view CMD)

To view the documentation of the code use command prompt and follow these steps

>pydoc

    'This command will help use to know extra details about this module and some arguments that we can pass and how these arguments effect the output'

```
pydoc - the Python documentation tool

pydoc-script <name> ...
    Show text documentation on something.    <name> may be the name of a
    Python keyword, topic, function, module, or package, or a dotted
    reference to a class or function within a module or module in a
    package.    If <name> contains a '\', it is used as the path to a
    Python source file to document. If name is 'keywords', 'topics',
    or 'modules', a listing of these things is displayed.

pydoc-script -k <keyword>
    Search for a keyword in the synopsis lines of all available modules.

pydoc-script -n <hostname>
    Start an HTTP server with the given hostname (default: localhost).

pydoc-script -p <port>
    Start an HTTP server on the given port on the local machine.    Port
    number 0 can be used to get an arbitrary unused port.

pydoc-script -b
    Start an HTTP server on an arbitrary unused port and open a Web browser
    to interactively browse documentation.    This option can be used in
    combination with -n and/or -p.

pydoc-script -w <name> ...
    Write out the HTML documentation for a module to a file in the current
    directory.    If <name> contains a '\', it is treated as a filename; if
    it names
```

1. Directly view the doc in command prompt
2. In browser (HTML)
3. Save the HTML file in local directory

6.3.1 Directly view in Command Prompt

dir>pydoc   <class or function name>

Eg:-

```
C:\Users\Jp227\Desktop\2048>pydoc random.shuffle
Help on method shuffle in random:

random.shuffle = shuffle(x, random=None) method of random.Random instance
    Shuffle list x in place, and return None.

    Optional argument random is a 0-argument function returning a
    random float in [0.0, 1.0); if it is the default None, the
    standard random.random will be used.
```

6.3.2 In browser (HTML)

dir>pydoc -b

Browser

6.3.3 Save the HTML File in Local Directory

dir > pydoc -w <Name of the python file>

It will create a html file with same python file name and exaltation in .html

7.   Programming Paradox

In every project a programmer will make decisions between time and perfection. Coding is complex and there are multiple ways to achieve the final result

That means two different programmers will choose different methods to solve the same problem there might be more widely changes in the approach , efficiency, merits etc···

In this project I am going to use Object Oriented Programming through python to design my application and I am going to divide the whole coding into 4 parts as follow

1. **Square   - which represents each box in board**
2. **Game Board - Board on which Squares arranged other features**
3. **Game   - Whole game stars from Game**
4. **Main   - Application starts from Main with main menu**

Going to use Container-ship

Square

Square objects 2D array(4x4) in GameBoard

Game Board

Game Board object in Game

Game

Game Object in Main

Main

The whole application starts from the object of the main Class
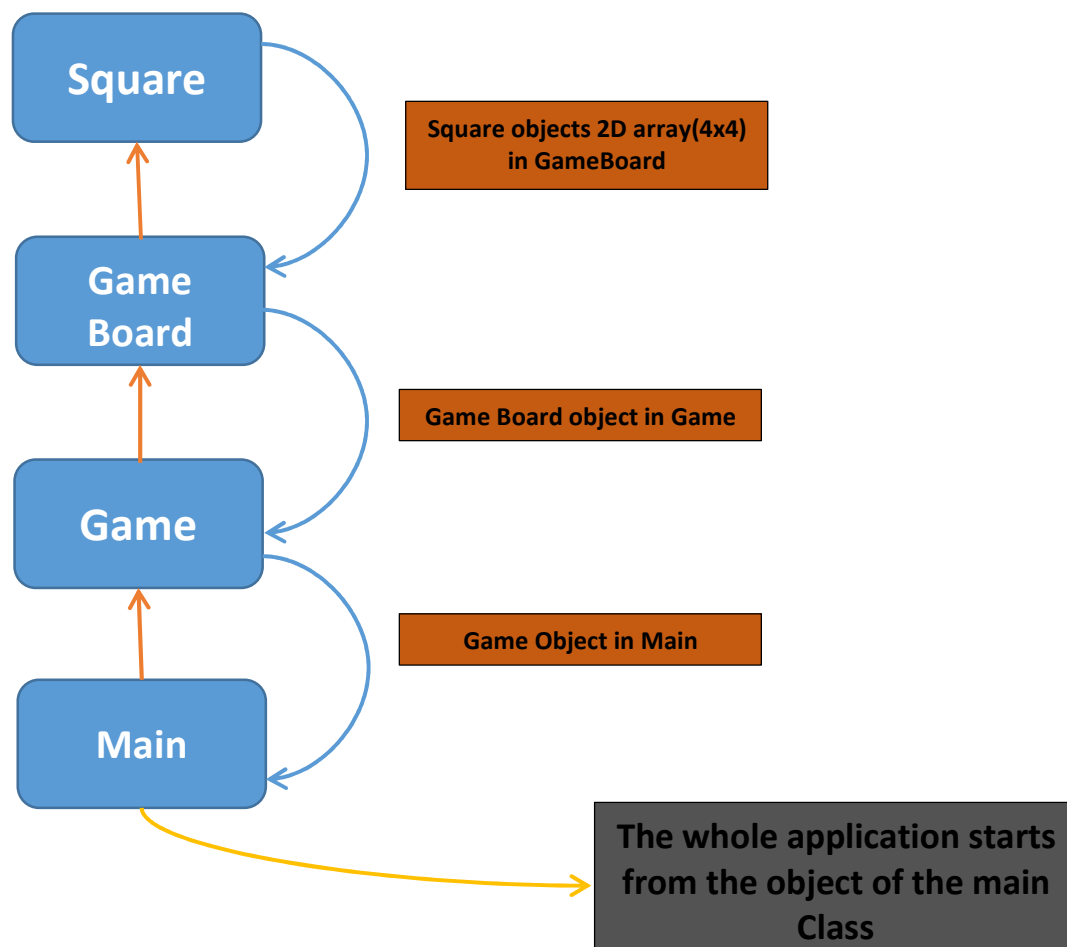
8.   Classes

8.1 Square

```
import pygame
pygame.init()
pygame.font.init()
class Square:
    """
        It is a class used to represent a square on gameboard
        = Data Members :-
        -    Value : which store value on square
        -    pos    : stores the loctaion of square on game board (i,j) 2D format
        -    text   : value in text format
        -    Color : color of the square

    """
    font = pygame.font.SysFont('comicsansms', 32)
    Display = None
    def __init__(self,pos):
        self.value = 0
        self.pos=pos
        self.text=None
        self.color=[255,255,255]
        self.update()

    def update(self,V=0):      # updating square (Number,Color) on square

    def colorUpdate(self):     # updating the color of the square

    def moveby(self,i,j,X,Y): # Animation for moving boxes
```

Output with basic code to run the object
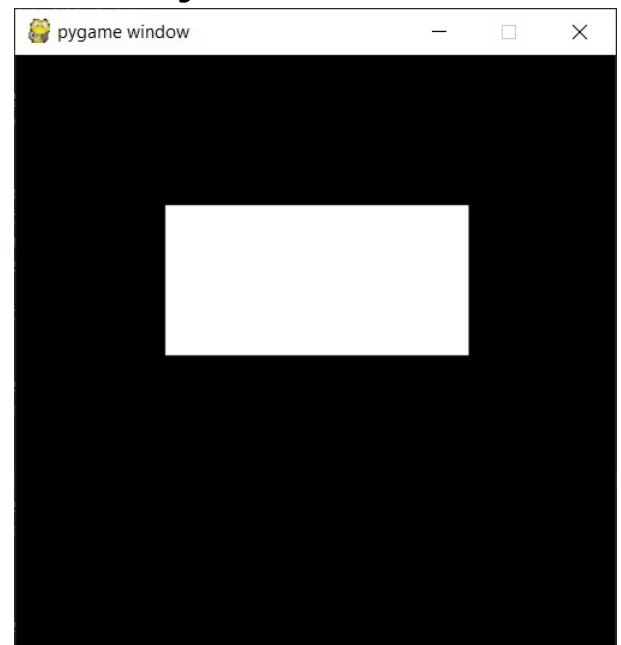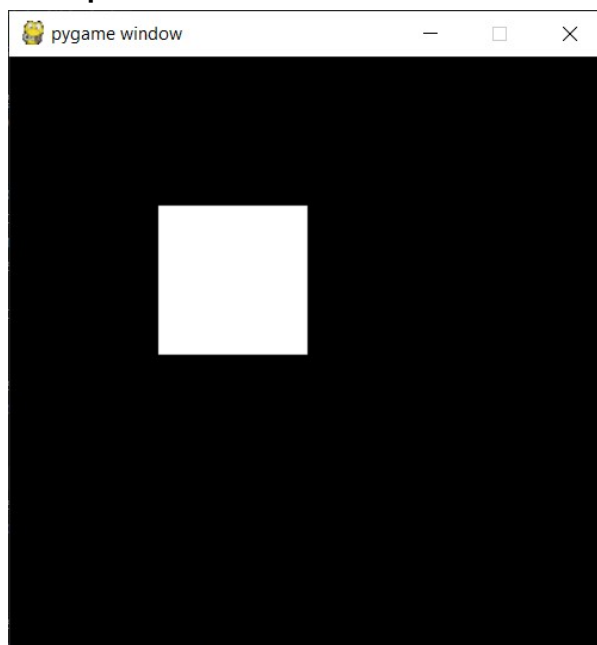
Output after pressing right key

8.2 Game Board

```
from numpy import array    # to create 4x4 matrix for gameboard
from square import Square   # to create objects for each square on gameboard
from random import choice   # for choosing a random postoin,values for inserting
new value into gameboard
import pygame
pygame.init()
class GameBoard:
    Direction={'up':(0,-2),'down':(0,2),'left':(-2,0),'right':(2,0)}
    #It is used to represent the game board
    #class variable:
    #Direction which is used to represent some parameters based on direction in modify() function
    #       data members:
    #             board    : stores all the square objects in 4x4 matrix format
    #             flag     : modification is done on board or not (True : done,False : not done)
    #             max      : stores the max value on board
    #             won      : Game Own or not
    #             gameOver : GameOver or not

    def __init__(self,Display):
    #         Requires a Display to show the square on gameBoard
    #          default value
    #          board = with 16 square objects with 0 as value and i,j values as position
    #          flag   = True
    #          max    = 2
    #          won    = False
    #          game over = False
    #         calling insert() function

    def array(self):
    #         Generating an 4x4 matrix only with values on the game board to store

    def restore(self,a):
    #         Regenerating the game board by taking values that are stored

    def display(self):
    #         this is used to display the game board

    def modify(self,event):
    #This used to rotate the game board virtually
    #   so that we can use same function for all inputs with small modification on gameboard
    #          rules followed for player inputs:
    #                 Up      : No change
    #                 Down    : Flip the rows
    #                 Left    : Transpose then Flip the columns
    #                 Right : Flip the rows then Transpose
```

```
#      After complication of updating the game board again
#                      Up      : No change
#                      Down   : Flip the rows
#                      Left    : Flip the columns then Transpose
#                      Right : Transpose then Flip the columns


    def motion(self,direction,List):
#   while modifying to give motion animation to the squares on board


    def shift(self,direction):
#   While modifying to remove the blank spaces in between squares based on players input


    def update(self,direction):
#   Update is used to initiate modification of squared
#          1. shifting the values will take place to remove middle spaces inbetween squares
#          2. merging will take place next and animation is give to it by using motion function


    def check(self):
#   To check whether Game Over or not by checking any two square are adjacent and same


    def insert(self):
#   To insert a new value on the game board at random position with either 2 or 4 randomly
```
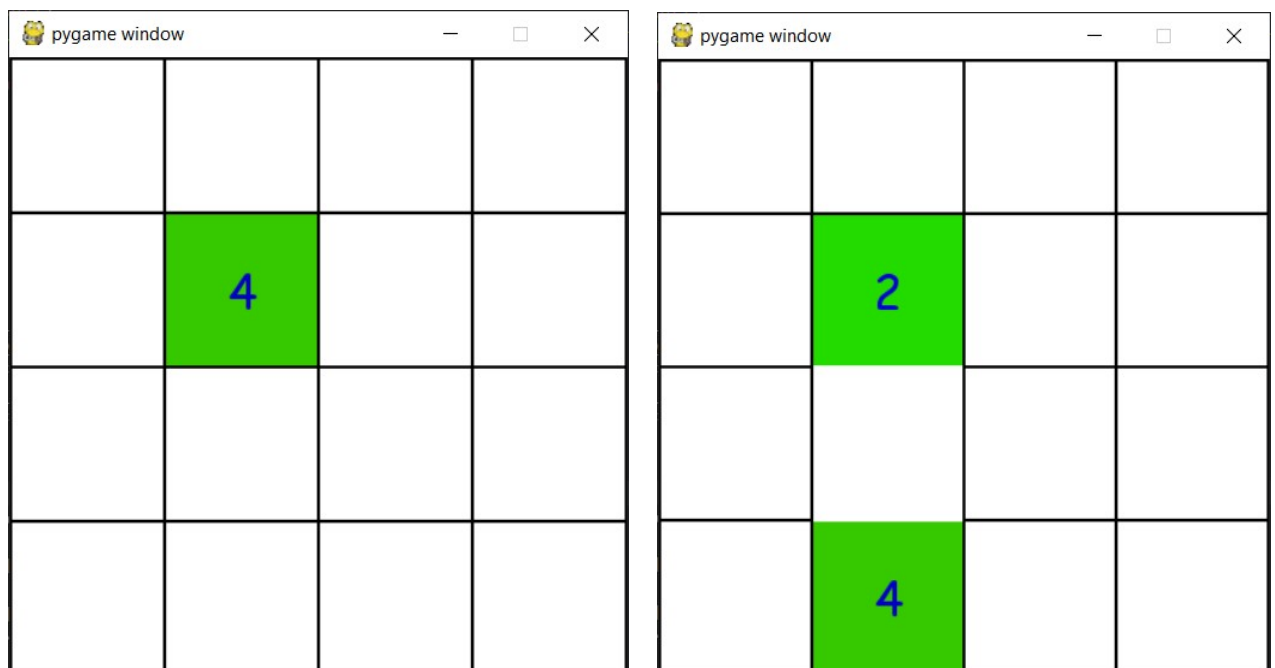
Output with basic code to run the GameBoard object

Output After pressing down key

8.3 Game

```
from gameboard import GameBoard
import pygame
import pickle
pygame.init()
class Game:
#       It acts as interface between game and player
#           like following :
#               1.input from keyboard
#               2.popup for asking to save /dont save game while closing in middle
#               3.pause the game

    def __init__(self,Display):
#           here we need a Display to show popup windows on screen
#               play           : it says about game is paused or playing
#               Game Board : to play game and display it
#               save           : it save text to display in popup
#               yes           :                    ''
#               no            :                    ''
#               x              : cross mark in popup

    def playgame(self):
#       This function will start game
#               Follow strings will be written by this function
#               1. Game Over   -   when Game Overs
#               2. Won         -    when play won the game by generating
#               3. resume      -   To resume the game if player wants to continue it
#               4. Exit        -    player wants to exit from game

    def pause(self):
#           This will pause the game and calls popup() function

    def popup(self):
#       This will generate popup with small menu
#               Save:
#                   yes
#                   no
#               cross available on popup to resume
#               String is written by this function
#               1.resume - if he/she wants to continue the game
#               2.Exit - if he/she wants to exit by saving/Not saving it
```

16

```
    def dontSave(self):
#      saving None value in the file if player don't want to save


    def yesSave(self):
# saving the matrix generated by gameboard.GameBoard.array()    function in Game-board
```

Outputs
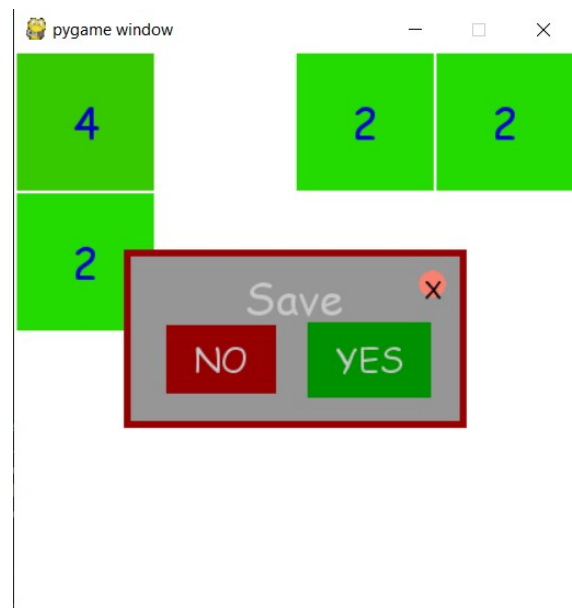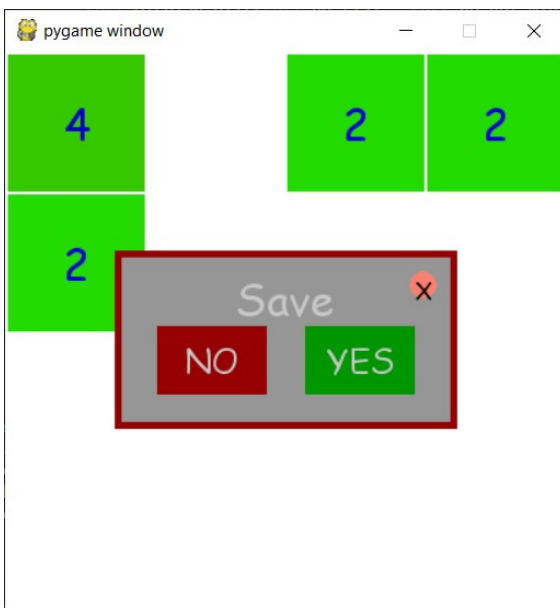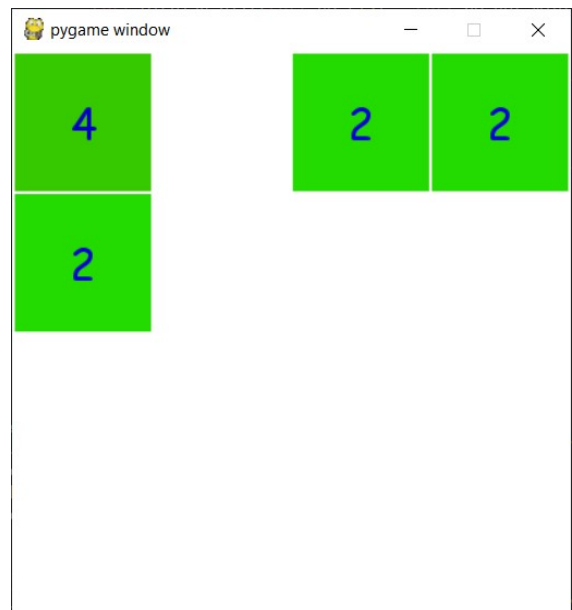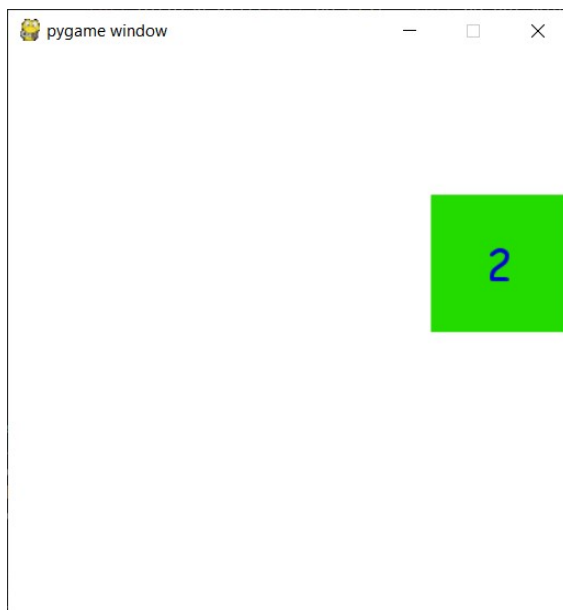
Outputs for popup window and while selecting "yes" in second image

8.4   Main

```python
import pygame
import pickle
from numpy import array
from random import choice
from game import Game
from gameboard import GameBoard
pygame.init()
pygame.font.init()
pygame.display.set_caption("2048")
icon = pygame.image.load('2048_IC5_2.ico')
pygame.display.set_icon(icon)

class Main:
#     It is the main class which handles all the menus like as followes:
#             1. Main Menu
#             2. Game Over Menu
#             3   Game Won Menu


#             Controls game
#             Loads old game/ new game based on inputs
    def __init__(self):
        self.Display=pygame.display.set_mode((410, 410))
        font=pygame.font.SysFont('comicsansms',35)
        self.Newgame=font.render('New Game',True,[32,27,29])
        self.Continue=font.render('Continue',True,[32,27,29])
        self.Exit=font.render('Exit',True,[32,27,29])
        font=pygame.font.SysFont('comicsansms',80)
        self.Won=font.render('U WON',True,[200,200,200])
        self.GameOver=font.render('Game Over',True,[200,200,200])
        self.Game=None


    def start(self):
#    After loading the game in Game variable This function is called It is # used to start
and handle the whole game by calling different function


# Based on the outputs given by the playgame() function other events    # are initianted


#             Game is controlled by single variable (running)
#                 True - continues
#                 False - Breaks
```

```python
    def mainmenu(self):
    #           main menu
    #                 Menu display while opening the game
    #                 1. Continue
    #                 2. New Game
    #                 3. Exit


        def resume(self):
    #         To reload the game from file to continue the previous game


        def gameover(self):
    #           Menu gets generated with two buttons
    #                 newgame
    #                 exit


        def won(self):
    #                 generates a menu if player won the game
    #                 1. New game
    #                 2. continue
    #                 we can use cross to exit from game


if __name__ == "__main__":
    #                 main is the object of Main class will handle whole game
    #                 creates a menu takes input
    #                 if player wants to exit from main menu exit
    #                 if not enter into game by using main.start() function
    main=Main()
    if main.mainmenu() != 'exit':
        main.start()
    pygame.quit()
```

Code Link:-

19

Outputs
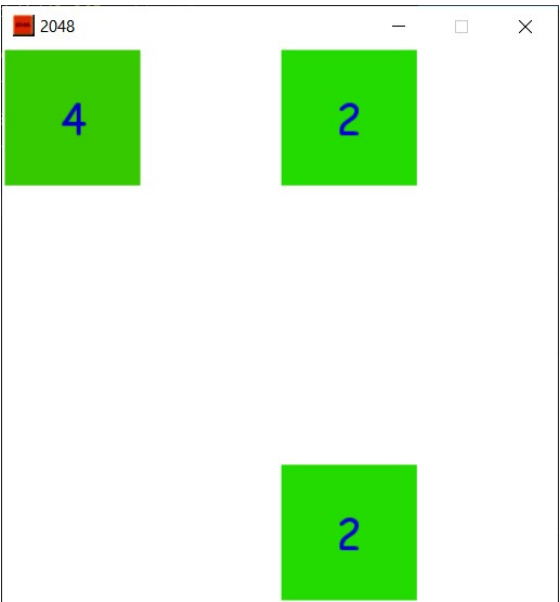
9.   Executable file Generation

From python code executable file can be generating by using pyinstaller package which is already discussed in *page 7*

To generate an exe file follow the below steps

Step 1:-   keep all the required files and the python files in one directory

Step 2:-   Make sure ,for which python file we need to generate exe file

Step 3:-   Open command Prompt

Step 4:-   Check whether pyinstaller in installed or not by using bellow command
`> pyinstaller`

If some output except error is printed on screen then pyinstaller is installed

Step 5:-   If not installed the use the below command to install
`>pip install pyinstaller`

Step 6:-   Go to the directory where your python file is located

Step 7:-   use the following command to generate Executable file
`>pyinstaller -F --noconsole    <filename>`

Step 8:-   Go to directory <dist> which is newly created after executing the above command there u can see the exe file

Note : To run the exe file it required some resource those are same that we used while coding like images, text files, etc...

Link to download the Executable file and its resources
https://github.com/Jyothi-prakash-muddana/2048-game/tree/master/Executable%20file

10. Setup File Generation

Inno Setup compiler which is going to bind the resource and the executable file at one location and help to install and uninstall option to the user to install our application in there PC and uninstall it by executing uninstall

Makes it easy for user and gives a grate look as an windows application

Inno Setup compiler can auto generate its own code

Follow the below steps :

Step 1:- Go to page 8 in this Document and use the link to install the Compiler
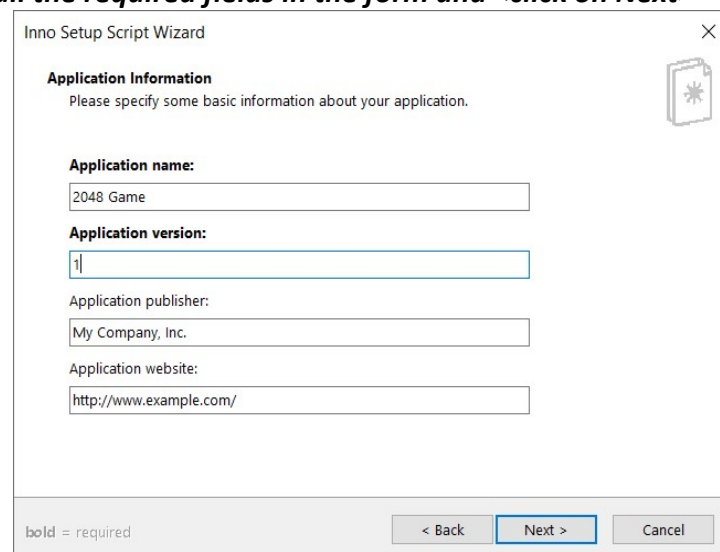
Step 2:- Open it

Step 3:- Choose Option   < create new empty Script using Script Wizard> and OK
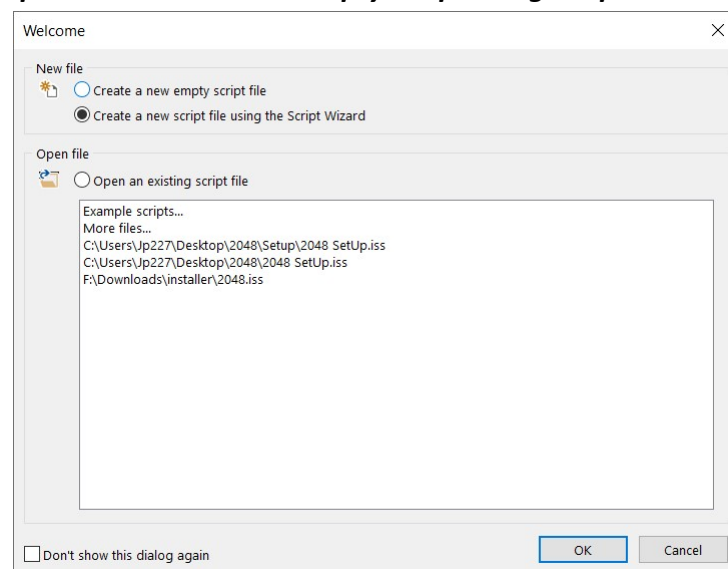
Step 4:- Just click <Next> (No changes at all)

Step 5:- Fill the all the required fields in the form and <click on Next>

Step 6 :- Click on Next again

Step 7 :- Add Main file/ other required files (paths) as shown in below image (Next)

Step 8:- (Next)

Setp 9: Add the Licence file(write something regarding your details in text file and use it as your Licence)   <Next>

Step 10:- complete all nexts till end <Next> and ok (save the file and run it)

Link for Setup File, inno code

https://github.com/Jyothi-prakash-muddana/2048-game/tree/master/Setup

11 . Install / Uninstall application

11.1   Installation of application

Follow below steps to install application in windows

Step 1: Download Setup file by using below link

https://github.com/Jyothi-prakash-muddana/2048-game/raw/master/Setup/mysetup.exe

Step 2: Double tap on that file to run it

Note :- This application is not licensed ,so if any issue with antivirus ignore it
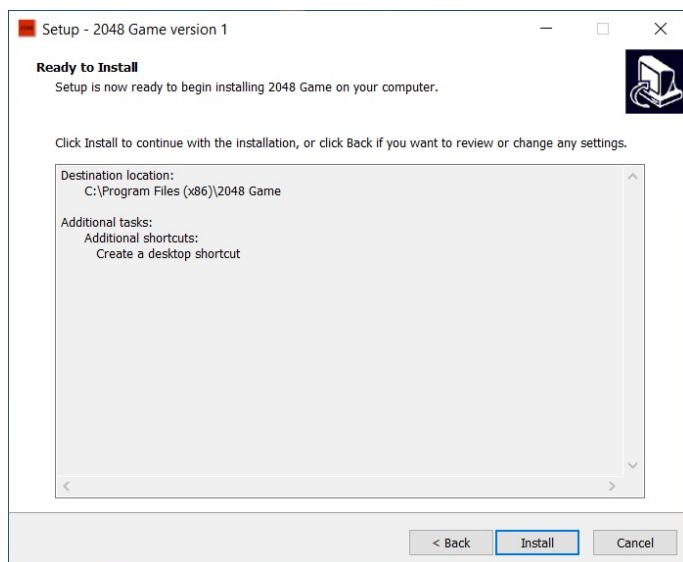
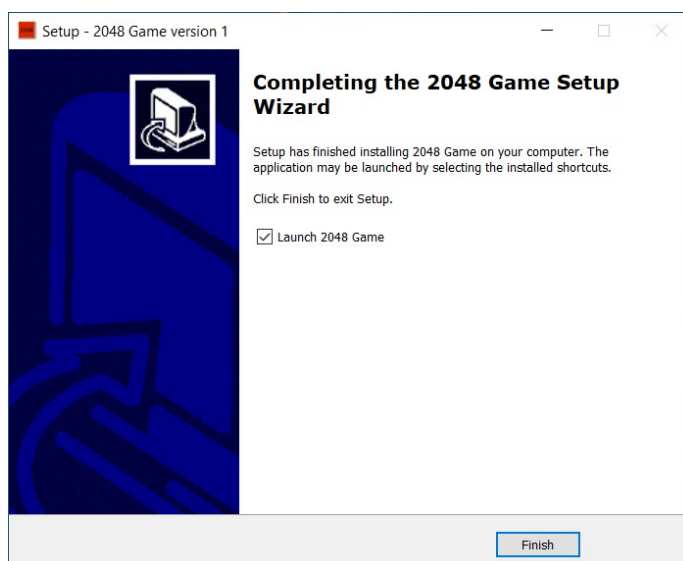Step 3: Accept the licence
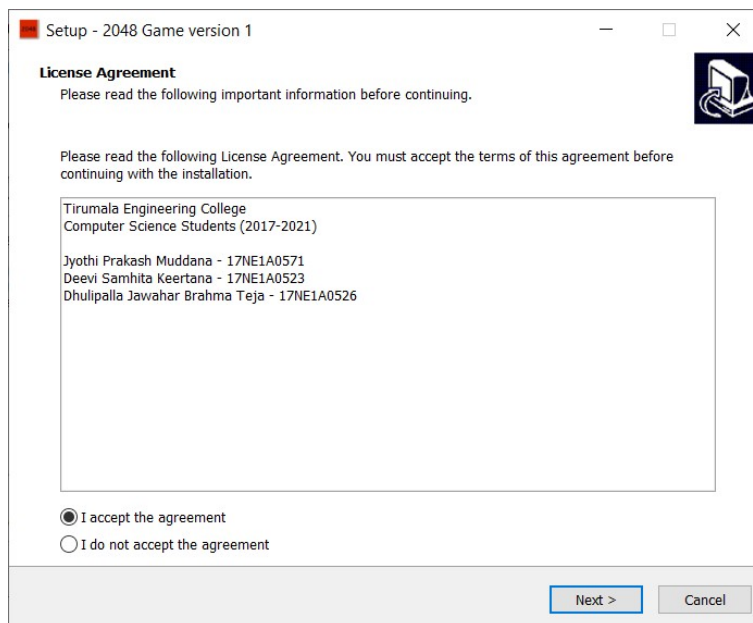
Step 4: Location to install and press Next
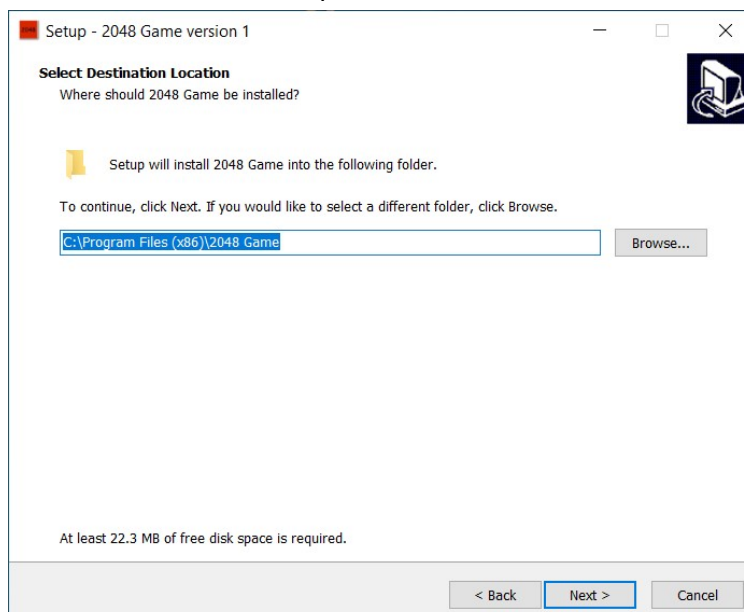
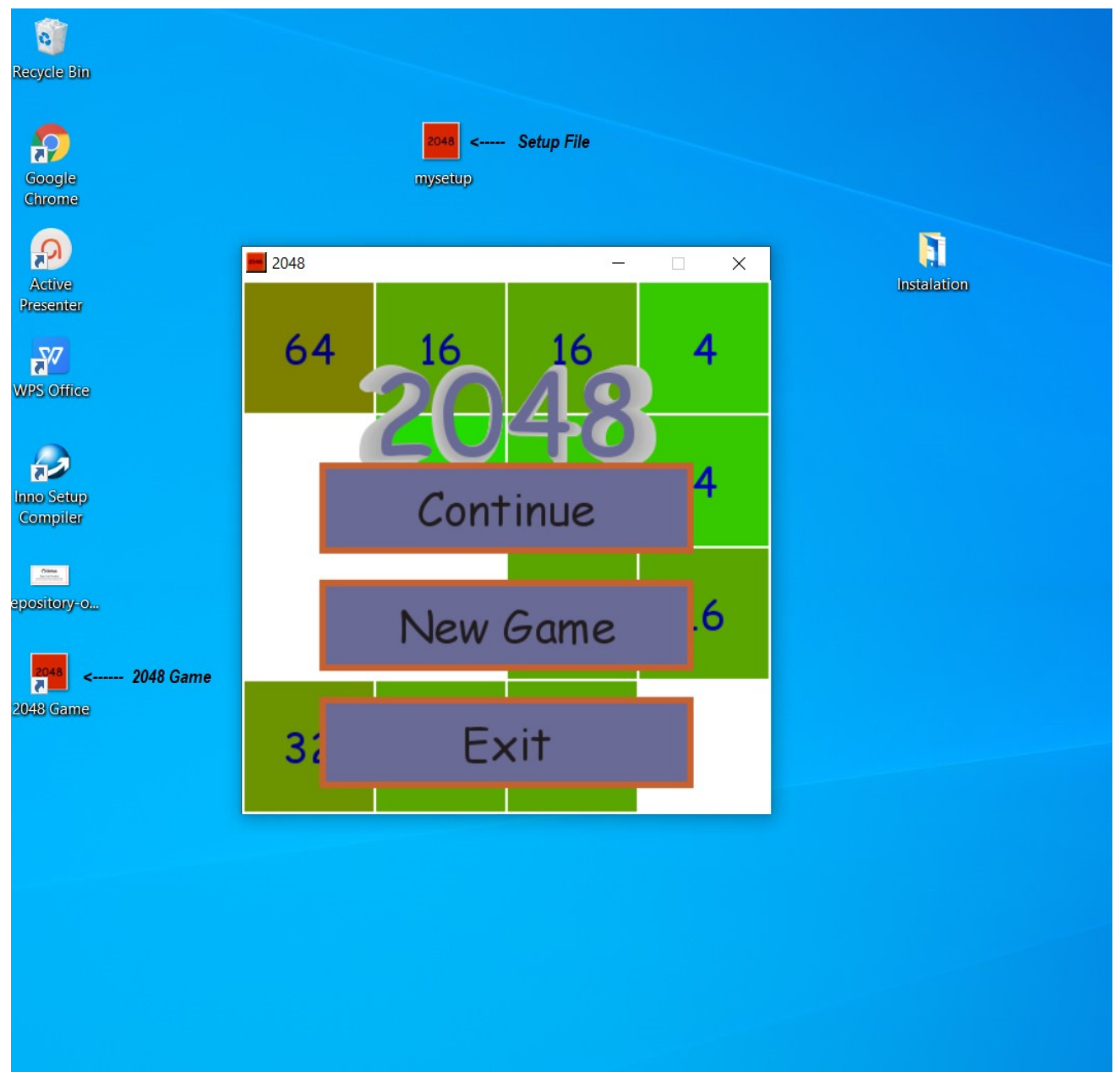Step 5: Check for creating a desktop shortcut and press Next
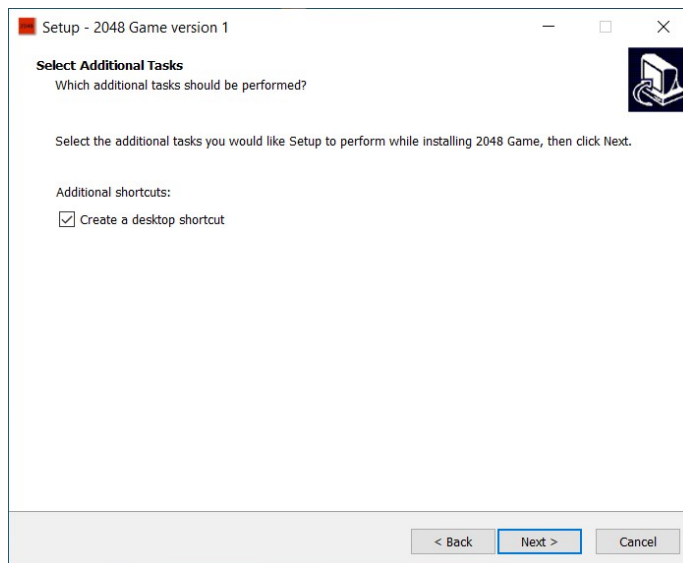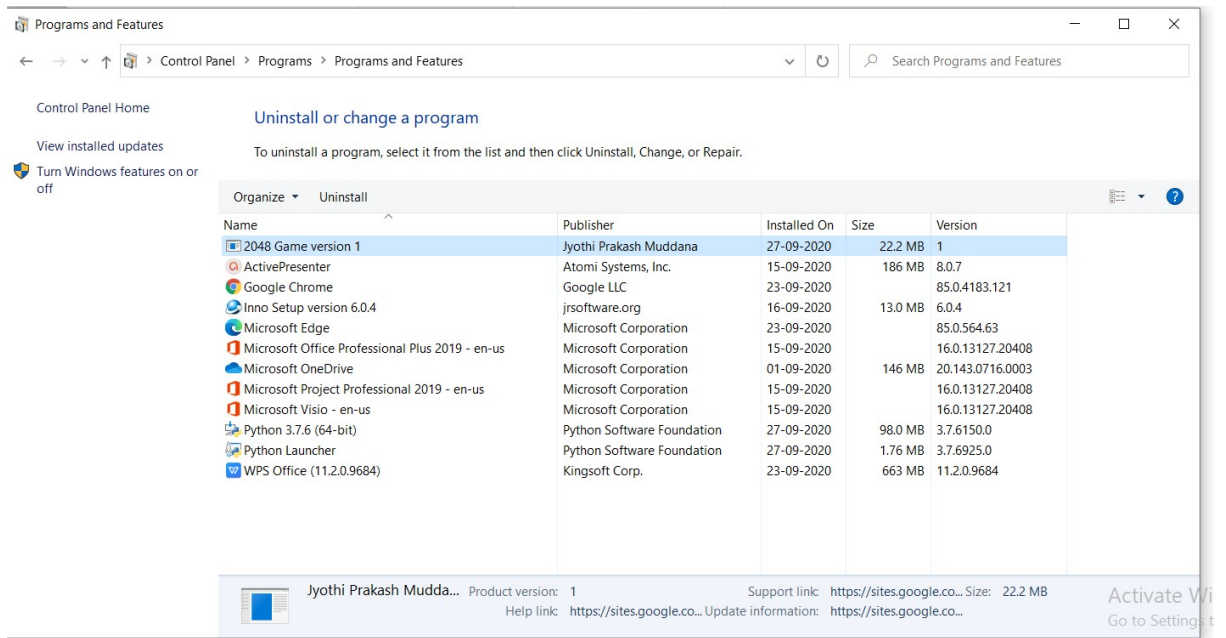

Step 6: Press Install

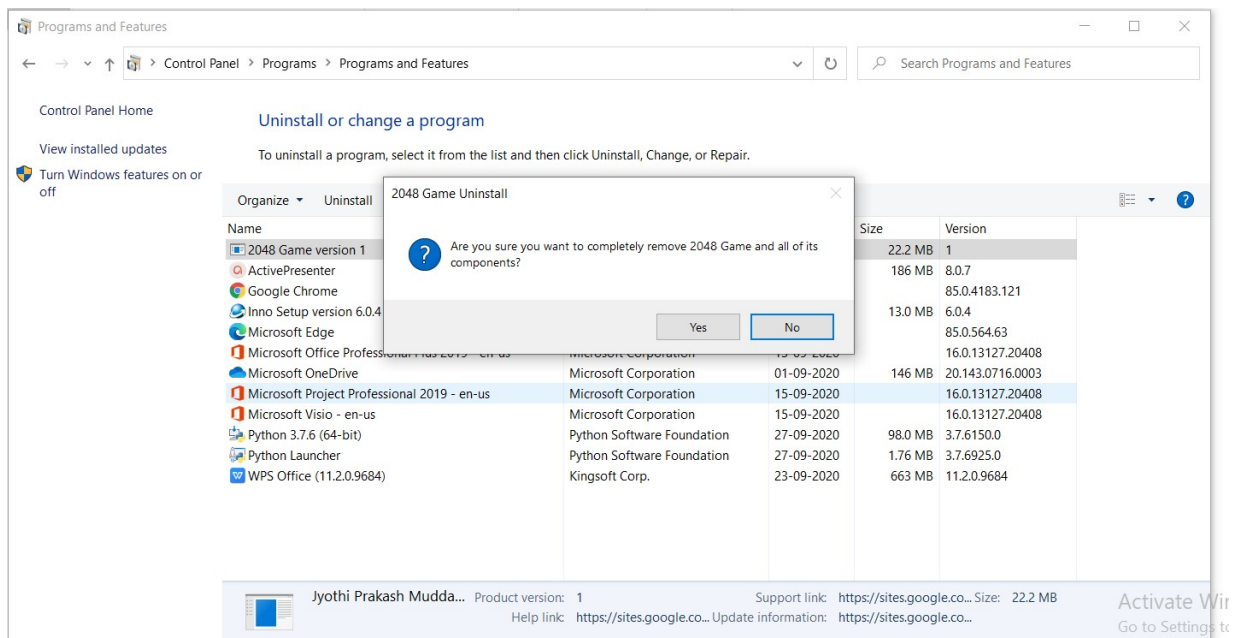Step 7: Finish

25

Step 8: Installed and running application

11.1   Uninstalling application
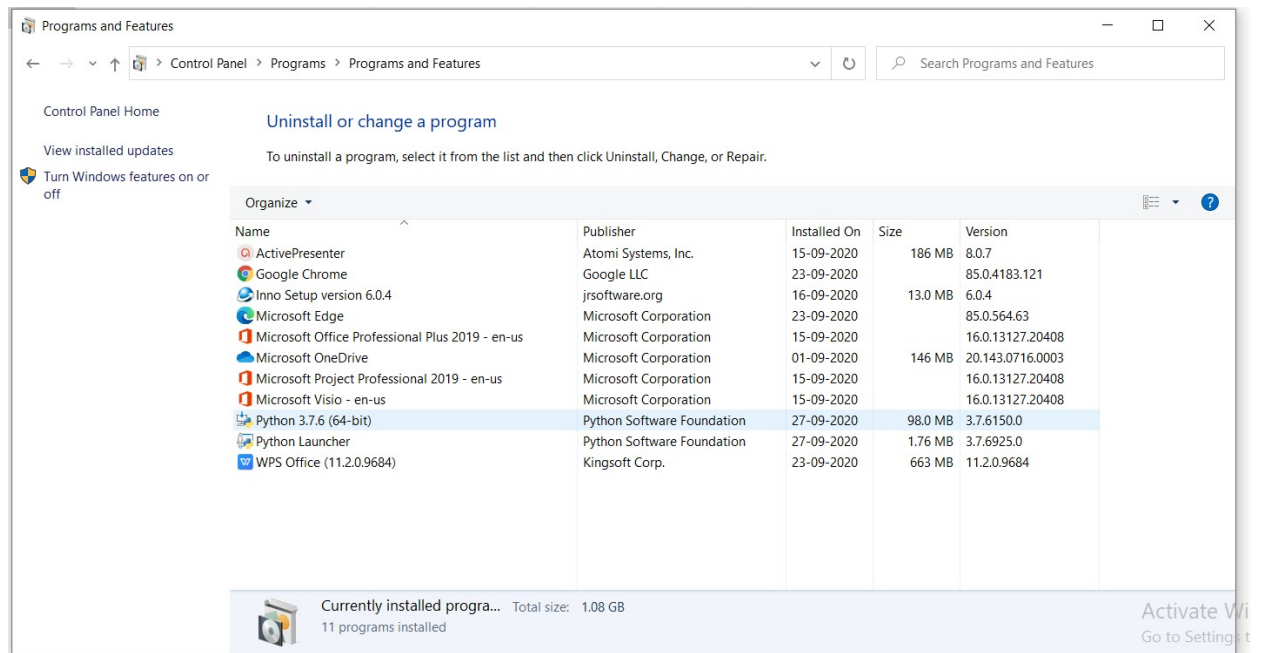
Steps to uninstall application

Step 1: Go to   [control Panel>Programs>Programs and Features]

Step 2: Select 2048 game and select Uninstall ,Press yes

Step 3: Press Ok and continue [application is uninstalled]

12. Bibliography

https://www.python.org/
https://www.pygame.org/news
https://numpy.org/
https://www.pyinstaller.org/
https://doughennig.com/Papers/Pub/InnoSetup.pdf
https://pypi.org/project/random2/
https://docs.python.org/3/library/pickle.html