**OSTPL Mini Project:**

**Snake Game**

**REPORT**

**A2 Batch**

**Neeraj Naik 1811027**

**Vatsal Pathak 1811033**

**Theory**

This project is designed with python, tkinter library and Pillow library for image handling**.**

**Python**

Python is a widely used general-purpose, high level programming language. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code. Python features a dynamic type system and automatic memory management. Python's simple syntax, modules and packages makes it possible for us to develop applications rapidly.

Professionally, Python is great for backend web development, data analysis, artificial intelligence, and scientific computing. Python contains a large library of standard functions which can be used for common programming tasks.Many developers have also used Python to build productivity tools, games, and desktop apps.The syntax of the language is clean and the length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

Python supports modules and packages, which encourages program modularity and code reusability. Python also supports file management and database connection. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

**tkinter**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

1. Import the Tkinter module.

2. Create the GUI application main window.

3. Add one or more of the above-mentioned widgets to the GUI application.

4. Enter the main event loop to take action against each event triggered by the user.

Tkinter Widgets:

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

1. **Tk(screenName=None, baseName=None, className=’Tk’, useTk=1):**  To create a main window, tkinter offers a method ‘Tk(screenName=None, baseName=None, className=’Tk’, useTk=1)’. To change the name of the window, you can change the className to the desired one. The basic code used to create the main window of the application is:  
   m=tkinter.Tk() where m is the name of the main window object

**mainloop():** There is a method known by the name mainloop() is used when your application is ready to run. mainloop() is an infinite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed. m.mainloop()

1. import tkinter m = tkinter.Tk() '''widgets are added here''' m.mainloop()

tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes.

1. **pack() method:**It organizes the widgets in blocks before placing in the parent widget.
2. **grid() method:**It organizes the widgets in grid (table-like structure) before placing in the parent widget.
3. **place() method:**It organizes the widgets by placing them on specific positions directed by the programmer.

Some of the major widgets are explained below:

1)**Button**:To add a button in your application, this widget is used.

The general syntax is:

w=Button(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* activebackground: to set the background color when the button is under the cursor.
* activeforeground: to set the foreground color when the button is under the cursor.
* bg: to set the normal background color.
* command: to call a function.
* font: to set the font on the button label.
* image: to set the image on the button.
* width: to set the width of the button.
* height: to set the height of the button.

Example

import tkinter as tk

r = tk.Tk()

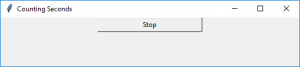
r.title('Counting Seconds')

button = tk.Button(r, text='Stop', width=25, command=r.destroy)

button.pack()

r.mainloop()

Output:



2)**Entry**:It is used to input the single line text entry from the user.. For multi-line text input, Text widget is used.The general syntax is:

w=Entry(master, option=value)

master is the parameter used to represent the parent window.

There are a number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* bd: to set the border width in pixels.
* bg: to set the normal background color.
* cursor: to set the cursor used.
* command: to call a function.
* highlightcolor: to set the color shown in the focus highlight.
* width: to set the width of the button.
* height: to set the height of the button.

Example:

from tkinter import \*

master = Tk()

Label(master, text='First Name').grid(row=0)

Label(master, text='Last Name').grid(row=1)

e1 = Entry(master)

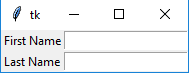
e2 = Entry(master)

e1.grid(row=0, column=1)

e2.grid(row=1, column=1)

mainloop()

Output:

****

3)**Scrollbar:** It refers to the slide controller which will be used to implement listed widgets.The general syntax is:

w = Scrollbar(master, option=value)

master is the parameter used to represent the parent window.

There are a number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* width: to set the width of the widget.
* activebackground: To set the background when the mouse is over the widget.
* bg: to set the normal background color.
* bd: to set the size of the border around the indicator.
* cursor: To appear the cursor when the mouse over the menubutton.

Example:

from tkinter import \*

root = Tk()

scrollbar.pack( side = RIGHT, fill = Y )

mylist = Listbox(root, yscrollcommand = scrollbar.set )

for line in range(100):

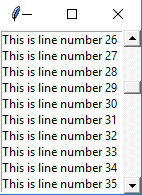
mylist.insert(END, 'This is line number' + str(line))

mylist.pack( side = LEFT, fill = BOTH )

scrollbar.config( command = mylist.yview )

mainloop()

Output:



**Project:**

This project creates the Snake Game using Tkinter to build skill in the use of Python GUI, Object-Oriented programming design and implementation, and data representation. Students will learn about Object-Oriented programming using existing Classes and Objects, as well in designing python based GUI applications.

Snake is an older classic video game. It was first created in late 70s. Later it was brought to PCs.

In this game the player controls a snake. The user controls a snake which continuously moves in the direction its’ head is facing. If the snake runs into one of the outer boundary walls or its’ own body, it will result in fatal injury to the snake and the player will lose the game as a result. An apple will always be spawned on the map, at a randomly generated location. The snake can eat apples by running over them. When the snake eats the apple that is on the map, the snake will grow 1 unit longer and another apple will appear in another randomly generated location. The goal of the game is to control the snake in such a way that it eats as many apples as possible (i.e. the snake grows as long as possible) without dying. This gets harder the larger the snake gets, because there will be less available space for the snake to move in its’ environment and the speed of the snake is increased, so the game gets very strategic as the snake approaches longer lengths. There are brief instructions and more on the application sheet.

**Development**

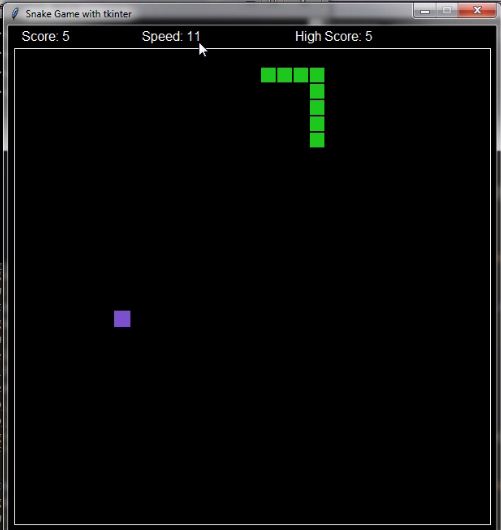
The size of each of the joints of a snake is 10 px. The snake is controlled with the cursor keys. Initially, the snake has three joints. The game starts immediately. When the game is finished, we display game over message with the score in the center of the window.

We use the Canvas widget to create the game. The objects in the game are images. We use canvas methods to create image items. We use canvas methods to find items on the canvas using tags and to do collision detection.



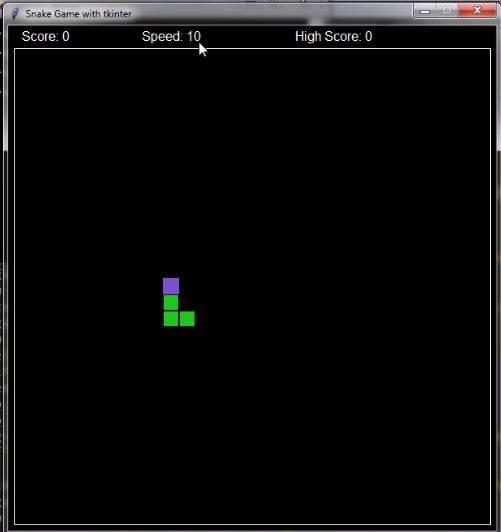
The initial window gives the option to user to start the game.

If the start button is selected , the tkinter window is cleared and a function is called.



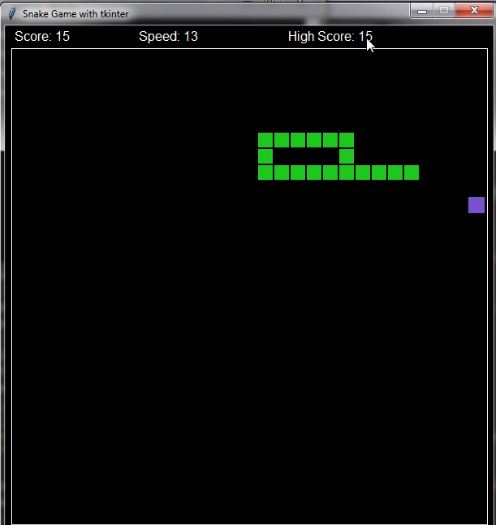
A new canvas is created and the images of the snake and food are loaded using Pillow library and the objects of snake and food are created.

The snake is made to change its position after a specified time and the direction is based on user inputs of the direction arrows.



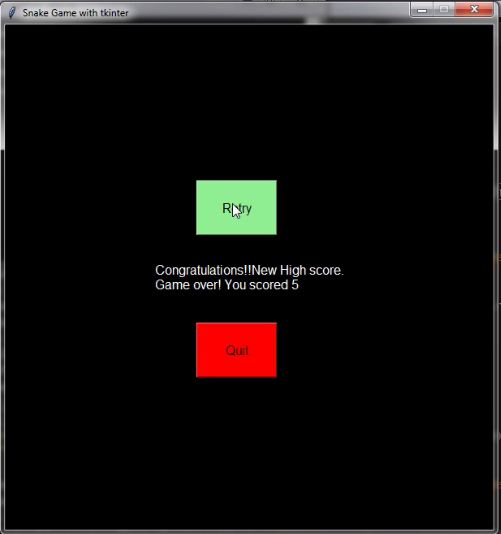
.Whenever the snake head encounters the food object, the length of the snake body increments by 1 alongwith the score.

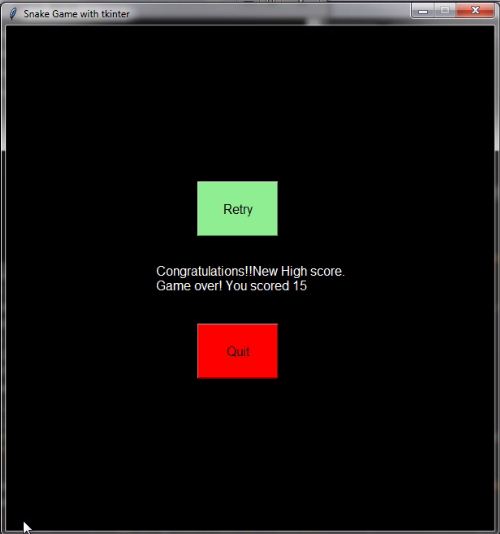
If the score reaches a multiple of 5 , the speed of the game increments.



.

Whenever the snake head collides with either its head or the boundaries of the game window, the game is over and the screen is cleared and a function end\_game is called.





The end\_game function makes either of the two messages appear.

If the score is highscore, it tells player that they have a new high score and if it is not ,then the high score and cureent score is informed.

Now the user can choose to either play again, or quit the game.

If he plays again, the previous high score is remembered but if he quits ,all data is lost.