

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

PROJECT 1.1

Turtle Library

TABLE OF CONTENTS

[1. Objectives 2](#_Toc140765801)

[1.1. Task A – Setup 3](#_Toc140765802)

[1.2. Task B – The Project 5](#_Toc140765803)

[1.3. Task C – Bonus 5](#_Toc140765804)

# Objectives

The turtle library in Python is a module that provides a way to create graphics and shapes by controlling a virtual turtle. The turtle is a cursor that can be moved around the screen and used to draw lines and shapes.

The Turtle Library provides a simple and fun way for beginners to learn about programming and computer graphics. It is often used in introductory programming courses to teach basic programming concepts such as loops, conditionals, and functions.

You can create drawings, animations, and even games with the turtle library. The library provides a variety of methods to control the turtle, such as forward, backward, left, right, penup, pendown, and many more. You can also change the color of the turtle, the pen size, and the screen’s background colour.

Overall, the turtle library in Python is an excellent tool for beginners to learn about programming and computer graphics, and it can be lots of fun to use.

In this first coding project, we will create an advanced turtle application. The project aims to research an extensive library, demonstrate branches, loops and functions and use additional libraries.

## Task A – Setup

import turtle

**Screen()** is a function that creates and returns a Turtle Screen object. The **Screen** object is a top-level object in the turtle library, and represents the drawing window or canvas where turtle graphics are displayed.

The **Screen** object provides several methods and attributes that allow you to customize the drawing window, such as setting the window size, title, and background color, and controlling whether or not the turtle is visible.

# create a Turtle Screen object

screen = turtle.Screen()

# set the window title and background color

screen.title("My Turtle Graphics")

screen.bgcolor("white")

We use a for loop in the code below to create a square.

# create a turtle object

turtle\_1 = turtle.Turtle()

# draw a square

for i in range(4):

    turtle\_1.forward(100)

    turtle\_1.right(90)

Now, let us create a simple, user-defined function to operate on our turtle to draw colored squares.

# ---function to draw colored squares with the global turtle turtle\_1 #

def draw\_square(size,color):

    turtle\_1.hideturtle()

    turtle\_1.penup()

    turtle\_1.goto(-size/2,size/2)

    turtle\_1.pencolor(color)

    turtle\_1.pendown()

    turtle\_1.pensize(4)

    for i in range(4):

        turtle\_1.forward(size)

        turtle\_1.right(90)

Set our colors

colors=[“red”, “orange”, “yellow”, “green”, “blue”]

Call our user-defined function five times using different colors and sizes

# ---Call to function with a for loop--- #

for i in range(6):

    draw\_square(i \* 100, colors[i-1])

Make sure to call a function on the Screen Object that enables closing the window:

**screen exitonclick()** or **screen.mainloop** are good candidates for this.

# This function keeps the window open until the user closes it

screen.mainloop()

## Task B – The Project

Create a turtle application that meets the following minimum requirements:

* Uses at least two control structures (branching, loops, etc.)
* Makes use of logical data structures like lists, tuples, sets and dictionaries (For example, you can create a list to hold all turtles, or you can use data structures to hold other application data)
* Makes use of at least two additional modules/libraries
* Makes use of at least two user-defined functions
* The code is well-documented with comments, and the coding style is consistent.

Research the turtle libraries and their full capabilities. Use the REPL / interactive interpreter to experiment with all the available methods and properties.

Share your knowledge with your team and brainstorm possible ideas for your turtle application based on the research in the previous step. Make sure the final idea meets the minimum requirements.

Break down your idea into parts and write pseudo code. Divide the coding tasks in your group.

Code the different parts of your turtle application and test the results.

Good luck!

## Task C – Bonus

The following are some of the bonus requirements that are set for the turtle application:

* Has some form of user interaction (like a game).
* Use advanced programming structures (like list comprehensions, lambda’s, etc.).
* Represents real data structures (like JSON, CSV, etc.).
* Makes use of own user-defined modules (modular code).
* Has a realistic use-case / is useful.