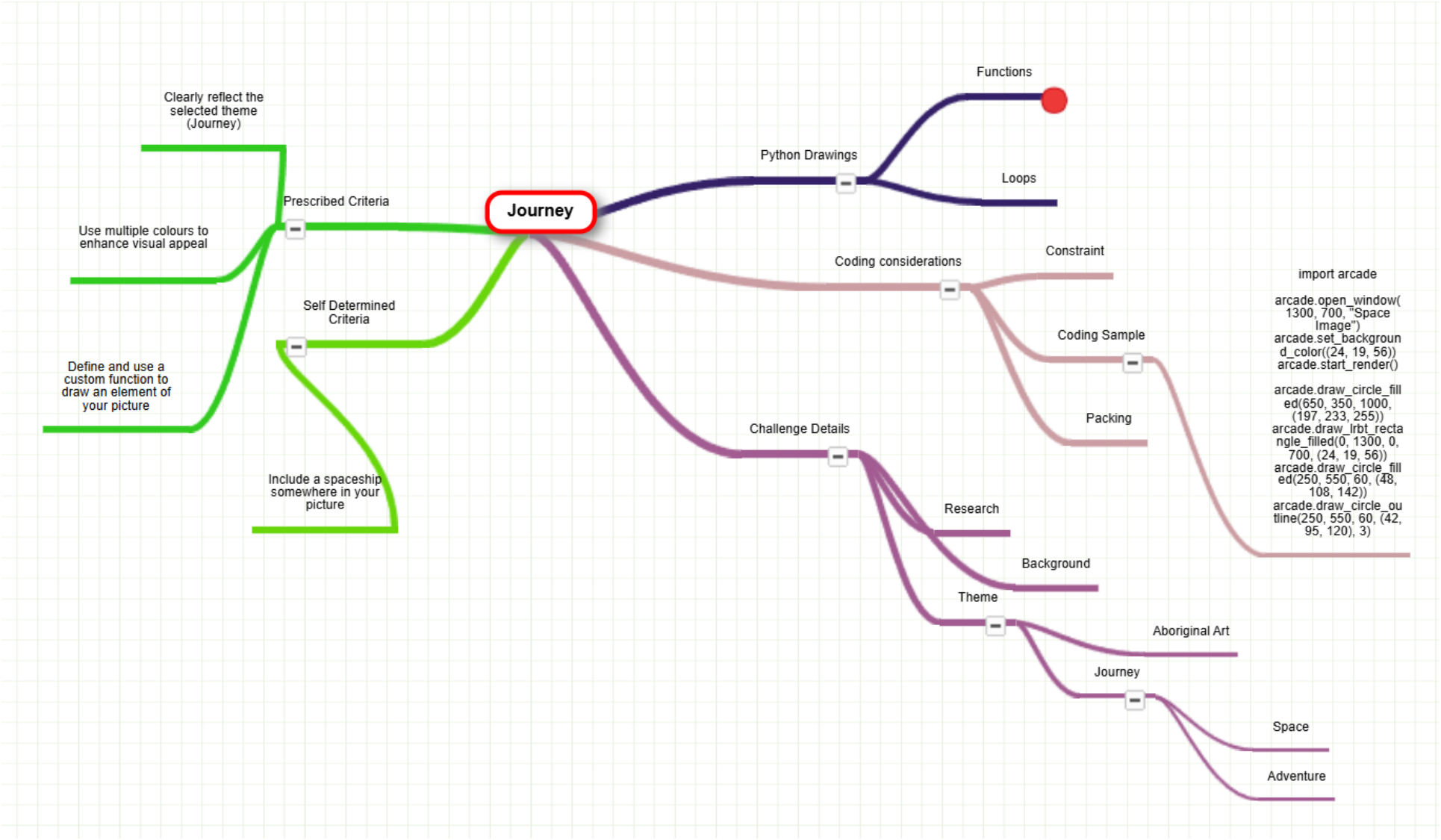
Student Name: TAMANG, Mehul

Year Level: 10

Contents

Mind Map


Journey
Prescribed Criteria
Clearly reflect the selected theme (Journey)
Use multiple colours to enhance visual appeal
Define and use a custom function to draw an element of your picture
Self Determined Criteria
Include a spaceship somewhere in your picture
Python Drawings
Functions
Loops
Coding considerations
Constraint
Coding Sample
Packing
import arcade
arcade.open_window(1300, 700, "Space Image")
arcade.set_background_color((24, 19, 56))
arcade.start_render()
arcade.draw_circle_filled(650, 350, 1000, (197, 233, 255))
arcade.draw_lrbt_rectangle_filled(0, 1300, 0, 700, (24, 19, 56))
arcade.draw_circle_filled(250, 550, 60, (48, 108, 142))
arcade.draw_circle_outline(250, 550, 60, (42, 95, 120), 3)
Challenge Details
Research
Background
Theme
Aboriginal Art
Journey
Space
Adventure

Pseudocode

START

DEFINE FUNCTION draw_spaceship(x, y):

DRAW a grey filled rectangle for the main body of the ship at position (x, y)

DRAW a grey filled triangle on top of the body at:

Left vertex: (x, y + 10)

Right vertex: (x + 100, y + 10)

Apex: (x + 50, y + 70)

DRAW a filled triangle in these locations in orange:

Left vertex: (x, y - 200)

Apex: (x + 50, y - 60)

Right vertex: (x + 100, y - 200)

DRAW a filled triangle in these locations in black:

Left vertex: (x, y - 10)

Right vertex: (x - 20, y - 40)

Apex: (x, y - 60)

DRAW a filled triangle in these locations in black:

Left vertex: (x + 100, y - 10)

Right vertex: (x + 120, y - 40)

Apex: (x + 100, y - 60)

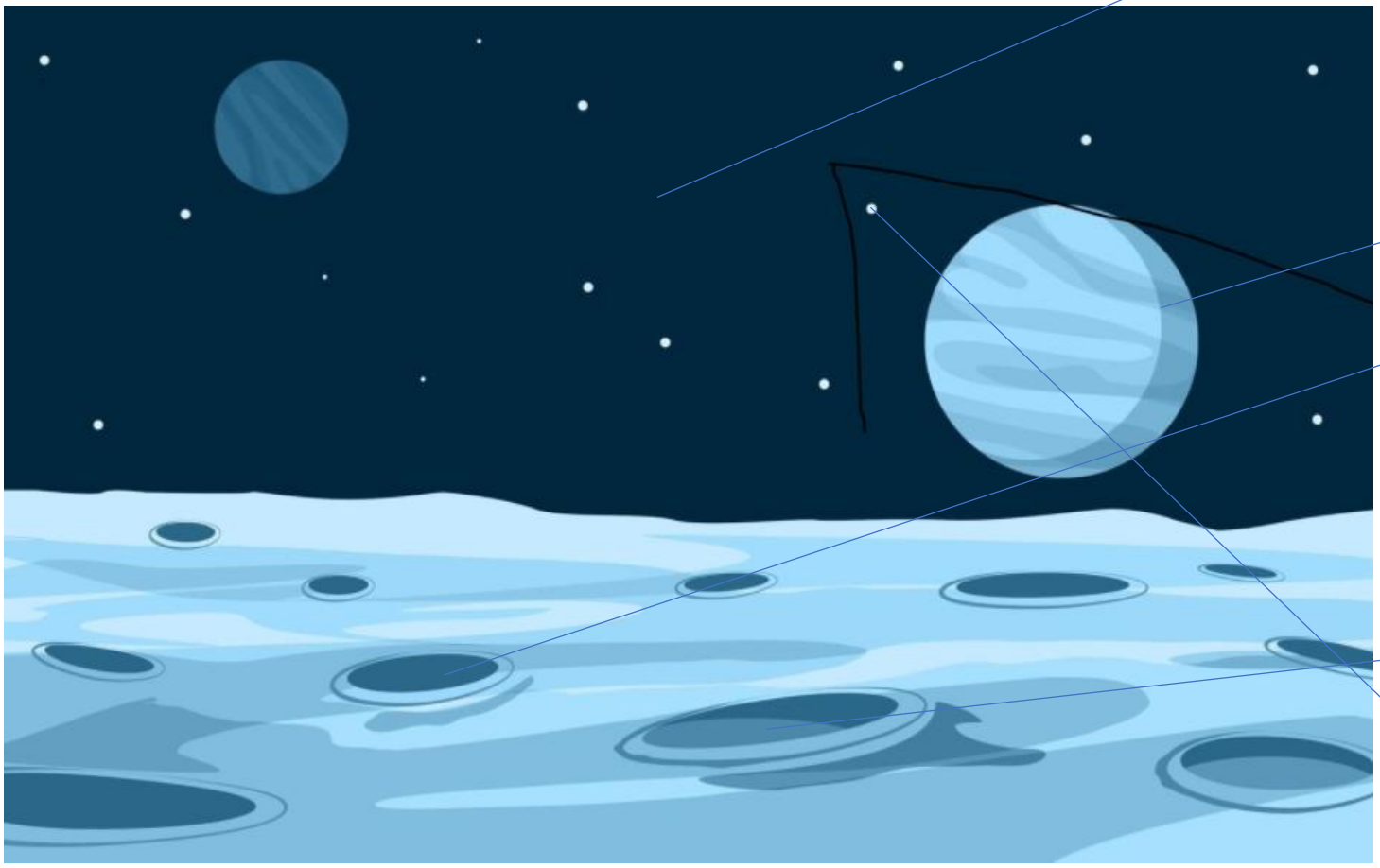
END FUNCTION

FOR star **IN** stars:

DRAW a circle

END

Annotated Sketches



RGB Colour of (24,19,56)

A planet which can be created using a circle shape primitive.

For the craters, I can use loop functions to repeat the craters.

```
craters = [
    (180, 180, 250, 50), (650, 160, 250, 50),
    (950, 160, 250, 50), (1100, 100, 150, 30),
    (150, 100, 100, 25), (350, 250, 120, 30),
]

for x, y, w, h in craters:
    arcade.draw_ellipse_filled(x, y, w + 8, h, (100, 139, 167))
    arcade.draw_ellipse_filled(x, y, w, h, (50, 83, 106))
    arcade.draw_ellipse_filled(x , y , 0.7*w, 0.7*h, (140, 192, 222, 120))
```

Craters can be created by using a for loop and can be aligned.

Stars can be created by importing random or creating many mini small circles

Annotated Code

```
import arcade

arcade.open_window(1300, 700, "Space Image")
arcade.set_background_color((24, 19, 56))
arcade.start_render()

arcade.draw_circle_filled(650, 350, 1000, (197, 233, 255))
arcade.draw_lrbt_rectangle_filled(0, 1300, 0, 700, (24, 19, 56))
arcade.draw_circle_filled(250, 550, 60, (48, 108, 142))
arcade.draw_circle_outline(250, 550, 60, (42, 95, 120), 3)

pattern_color1 = (42, 95, 120, 100)
arcade.draw_line(220, 540, 280, 560, pattern_color1, 4)
arcade.draw_line(230, 560, 270, 580, pattern_color1, 4)
arcade.draw_line(265, 535, 230, 525, pattern_color1, 4)

arcade.draw_circle_filled(1050, 450, 100, (142, 190, 230))
arcade.draw_circle_outline(1050, 450, 100, (108, 140, 177), 4)

overlay_color = (108, 140, 177, 120)
arcade.draw_line(930, 420, 1170, 480, overlay_color, 10)
```

Importing Arcade: The arcade library is imported to provide functions for creating graphics.

Opening a Window: A window is created with a width of 1300 and a height of 700, named "Space Image".

Setting Background Colour: The background colour is set using RGB values.

Starting Render Process: The rendering process begins, preparing for drawing shapes.

Drawing Shapes: Several shapes are drawn:

- A large, filled circle in the centre, representing a celestial body.
- A rectangle that covers the entire window, matching the background colour.
- A smaller filled circle and an outlined circle, representing a planet or moon.

```
stars = [
    (50, 650), (200, 620), (1200, 680), (1250, 610),
    (900, 600), (1150, 540), (700, 520), (450, 500),
    (1100, 300), (1150, 350), (750, 200)
]
small_stars = [
    (70, 620), (150, 230), (1045, 980), (1000, 690),
    (870, 550), (350, 40), (600, 760), (320, 600),
    (980, 160), (1790, 950), (750, 220)
]

for star in stars:
    arcade.draw_circle_filled(star[0], star[1], 5, (242, 244, 245))

for star in small_stars:
    arcade.draw_circle_filled(star[0], star[1], 2, (179, 179, 179))

arcade.draw_ellipse_filled(300, 200, 450, 70, (168, 206, 234))
arcade.draw_ellipse_filled(650, 150, 600, 90, (140, 192, 222))
arcade.draw_ellipse_filled(1000, 180, 450, 70, (168, 206, 234))

arcade.draw_lrbt_rectangle_filled(0, 1300, 0, 300, (198, 232, 255))
```

Coordinates: The tuples in the stars and small_stars lists represent the x and y positions for drawing stars on the screen.

Drawing Functions:

- arcade.draw_circle_filled(x, y, radius, color): Draws a filled circle at the specified coordinates with a given radius and color.
- arcade.draw_ellipse_filled(x, y, width, height, color): Draws a filled ellipse (like clouds) at the specified position with given dimensions and color.
- arcade.draw_lrbt_rectangle_filled(left, right, bottom, top, color): Draws a filled rectangle defined by the left, right, bottom, and top edges.

```
craters = [
    (180, 180, 250, 50), (650, 160, 250, 50),
    (950, 160, 250, 50), (1100, 100, 150, 30),
    (150, 100, 100, 25), (350, 250, 120, 30),
]

for x, y, w, h in craters:
    arcade.draw_ellipse_filled(x, y, w + 8, h, (100, 139, 167))
    arcade.draw_ellipse_filled(x, y, w, h, (50, 83, 106))
    arcade.draw_ellipse_filled(x , y , 0.7*w, 0.7*h, (140, 192, 222, 120))
```

Craters List: Each tuple in the craters list contains the x and y coordinates, along with the width (w) and height (h) of the crater.

Drawing Functions:

- arcade.draw_ellipse_filled(x, y, width, height, color): This function is used to draw filled ellipses. In this context, it creates the visual representation of the craters.
- The first call to draw_ellipse_filled draws a slightly larger outer edge for the crater, giving it a defined border.
- The second call draws the main body of the crater with its specified width and height.

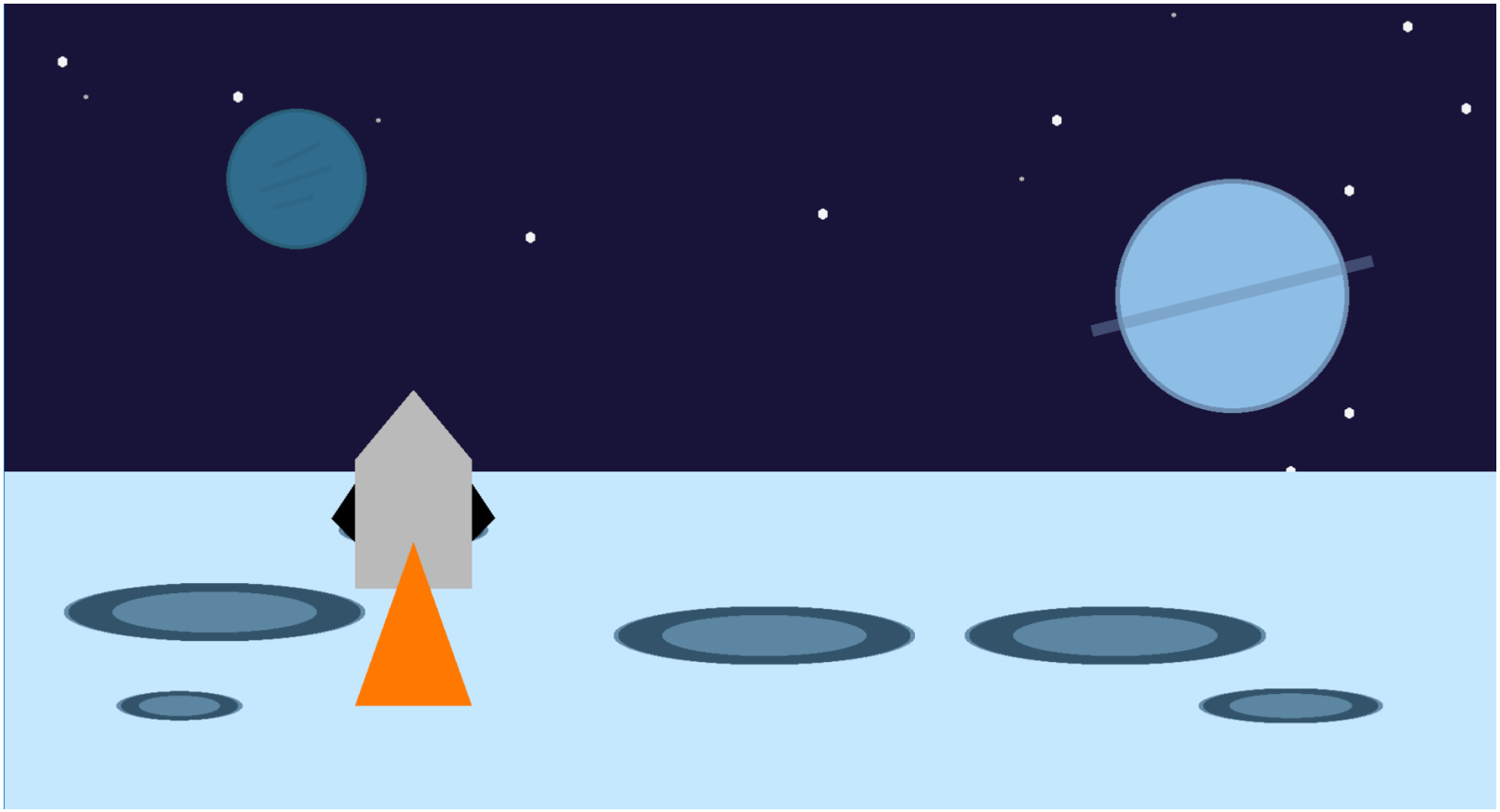
```
def draw_spaceship(x, y):

    arcade.draw_lrbt_rectangle_filled(x - 50 + 50, x + 50  + 50, 200, 310, (186, 186, 186))


    arcade.draw_triangle_filled(x - 50 + 50, y + 10, x + 50 + 50, y + 10, x + 50, y + 70, (186, 186, 186))


    arcade.draw_triangle_filled(x - 50 + 50, y - 200, x + 50, y - 60, x +50 + 50, y - 200, (255, 120, 0))


    arcade.draw_triangle_filled(x - 50 + 50, y - 10, x - 70 + 50, y - 40, x - 50 + 50, y - 60, (0, 0, 0))


    arcade.draw_triangle_filled(x + 50 + 50, y - 10, x + 70 + 50, y - 40, x + 50 + 50, y - 60, (0, 0, 0))


draw_spaceship(300, 300)

arcade.finish_render()
arcade.run()
```

Function Definition: def draw_spaceship(x, y): This function takes two parameters, x and y, which specify the position where the spaceship will be drawn.

Drawing Functions:

- arcade.draw_lrbt_rectangle_filled(left, right, bottom, top, color): Draws a filled rectangle representing the main body of the spaceship.
- arcade.draw_triangle_filled(x1, y1, x2, y2, x3, y3, color): This function draws filled triangles to create different parts of the spaceship, such as the nose, wings, and engine.

Final Screen Shots



In this short story Mehul, goes on journey aboard his spaceship, Odyssey as he travels the vastness of space. The narrative captures his sense of wonder and adventure as he fly's through a starry expanse, filled with distant celestial bodies, planets and craters.

`

Testing Table

Input	Expected Outcome	Actual Outcome	Comment
Run the program	A window opens displaying a space scene with various shapes and colours.	A window opens displaying a space scene with various shapes and colours.	The program runs successfully without any errors.
arcade.draw_circle_filled(250, 550, 60, (48, 108, 142))	A filled circle with radius 60 is drawn at (250, 550) in color (48, 108, 142).	A filled circle with radius 60 is drawn at (250, 550) in colour (48, 108, 142).	The circle is drawn correctly with the specified colour.
arcade.draw_circle_filled(1045, 980, 2, (179, 179, 179))	A small star (radius 2) is drawn at (1045, 980).	No star is visible at (1045, 980).	The coordinates exceed the window dimensions (1300x700), causing the star not to appear. This was corrected by adjusting the coordinates to be within the window bounds.
for x, y, w, h in craters: arcade.draw_ellipse_filled(x, y, w + 8, h, (100, 139, 167))	Craters are drawn as layered ellipses at specified coordinates.	Craters are drawn correctly, creating a 3D effect with shadows and highlights.	The layering of three ellipses for each crater successfully creates a sense of depth.
arcade.draw_triangle_filled(x - 50 + 50, y - 200, x + 50, y - 60, x + 50 + 50, y - 200, (255, 120, 0))	A triangle representing the spaceship's bottom is drawn correctly.	The triangle is drawn, but the base may not be visually appealing.	The coordinates for the triangle may need adjustment for better alignment.

User Survey

Success Criteria		User 1 (Yes/No)	User 1 Justification	User 2 (Yes/No)	User 2 Justification
Prescribed Criteria 1	Clearly reflect the selected theme (Journey)	Yes	The spaceship on the planet looks like it's about to take off into space. It gives a feeling of starting an adventure.	Yes	It's clearly a space journey. You see the ship and the stars and planets, which makes you think about travelling through space.
Prescribed Criteria 2	Use multiple colours to enhance visual appeal	Yes	I like the colour choices. The dark blue for space and the light blue for the ground make a good contrast. The orange flame on the rocket stands out well.	Yes	The colours look good. It's not too bright or messy. The different shades of blue for the craters make them look more realistic.
Prescribed Criteria 3	Define and use a custom function to draw an element of your picture	Yes	The spaceship is a key part of the picture, and I can see from the code annotations that it was made with a function, which is cool.	yes	The code seems well-organized by having the spaceship drawn with a function. It makes sense to group all those shapes together.
Self-Determined Criteria 1	Include a spaceship somewhere in your picture	Yes	The spaceship is right there in the foreground. It's the first thing I noticed.	Yes	The spaceship is clearly visible and looks like it's ready to launch from the planet's surface.

Evaluation

Criteria	Description	Requirements	Y/N	Justification	Recommendation
Prescribed Criteria 1	Clearly reflect the selected theme (Journey)	The image visually represents a journey about to begin.	Y	The image shows a spaceship on a planet's surface, looking out at a vast starry sky, symbolizing the beginning of a journey. A user survey confirmed this idea.	To further make this better, I could add animations like a pulsing engine light or slowly drifting stars to create a more sense of anticipation for the journey.
		The setting suggests exploration and movement from one place to another.	Y	The solid ground in the front contrasts with the endless space behind, representing the transition from a familiar place to an unknown destination.	In a future, I could add a faint, glowing nebula in the distance to serve as a visual goal or destination, making the idea of the journey more explicit.
Prescribed Criteria 2	Use multiple colours to enhance visual appeal	A palette of multiple, distinct colours is used.	Y	The colour palette includes dark navy, various shades of cyan and blue, white, grey, black, and a bright orange accent, creating a rich visual experience.	The palette is heavily blue-dominant. Including a contrasting colour such as a deep red or purple for one of the distant planets could add more detail and make it better.
		Colours are used to create contrast and depth.	Y	There’s a strong contrast between the dark space and the bright ground. The craters appear three-dimensional thanks to three different shades of blue layered together.	The sense of depth could be improved by applying principles of atmosphere, such as making the most distant stars slightly smaller and dimmer than those lower to the ground.
Prescribed Criteria 3	Define and use a custom function to draw an element of your picture	A custom function is defined using Python's def syntax.	Y	The code defines a function called draw_spaceship(x, y): that includes all the commands needed to draw the spaceship.	The function could be made better by adding parameters for size or colour, allowing for the creation of varied spaceships with a single function code.

		The defined function is called within the program to generate the element.	Y	This function is used in the main part of the program with draw_spaceship(300, 300), which successfully places the spaceship at those coordinates on the canvas.	The implementation is correct. No recommendation for change is needed.
Self – Determined Criteria	Include a spaceship somewhere in your picture	A recognisable spaceship is present in the final image.	Y	The image features a recognizable spaceship made from geometric shapes like rectangles and triangles, forming its body, nose cone, fins, and engine flame.	The current design is quite simple. For a more advanced version, I would design a more complex spaceship using more various shapes such polygon.
		The spaceship is logically positioned within the scene's context.	Y	The spaceship is positioned on the planet's surface, which makes sense for a code that is about to start its journey into space.	To better the spaceship into its environment, a simple shadow could be drawn underneath it which would ground it more realistically on the planet's surface.

Evaluation Summary

My solution was successful because it met all the prescribed and self-determined criteria outlined in the task sheet. Specifically, the final image clearly reflects the "Journey" theme (Prescribed Criteria 1) by visually narrating the start of an adventure, which was validated by user feedback. The project demonstrates strong aesthetic choice using a multi-coloured palette that creates contrast and depth (Prescribed Criteria 2). The code is well-structured implementing a custom function to generate a key visual element (Prescribed Criteria 3) and fulfills the self-determined of including a spaceship (Self-Determined Criteria 1). Overall, the project was a success.

References

Surface, M. (2019, April 22). *Cartoon Moon landscape with craters on space with stars, fantastic...* IStock. https://www.istockphoto.com/vector/cartoon-moon-surface-landscape-background-gm1144328145-307601409

Google. (2025). *Gemini*. Gemini.google.com; Google. https://gemini.google.com/app

PYTHON (2025). Python. [online] Python.org. Available at: https://www.python.org/