Test Plan

Zerg Mining Expedition

Josh Carroll, James Viner & Raymone Miller

Tool Developer Qualification Course
UMBC
March 7th, 2023

Test Plan, Mining

The program is designed to be an API which will help a developer in coding an actual program. This API will consist of Zerg overlords and Zerg Drones. The overlords will send drones out upon mining expeditions to collect and bring back minerals that the Zerg race need to survive. As these drones are deployed and explore the map they will suggest a route that is the best path to the minerals according to the drone's current knowledge of the map.

Instructor provided Files

- **/timeout.py**
  Defines functions needed for a program timeout.
- **/map.py**
  defines location, map, mineral context, and drone context classes.
- **/begin_mining_expedition.py**
  attempts to run a mining expedition w/ data types needed.
- **/sample_classes.py**
  defines dashboard, Drone and Overlord classes.
- **/maps**
  contains test maps.

Sample Files

- **None yet listed**

TC1:Installation

1. *git clone git@git.umbc.tc:tdqc/tdqc12/carroll/mining.git*
2. *cd mining*
   Expected:mining directory is created

prerequisites: In mining directory

TC2:Correct Branch

1. *git branch*
   Expected:main is default branch

Common Test Cases

TC3: Initialize a Dashboard

prerequisites: User's code must initialize a dashboard.

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop

from mining.GUI.dashboard import Dashboard

root = Tk()

example = Dashboard(root)

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user.

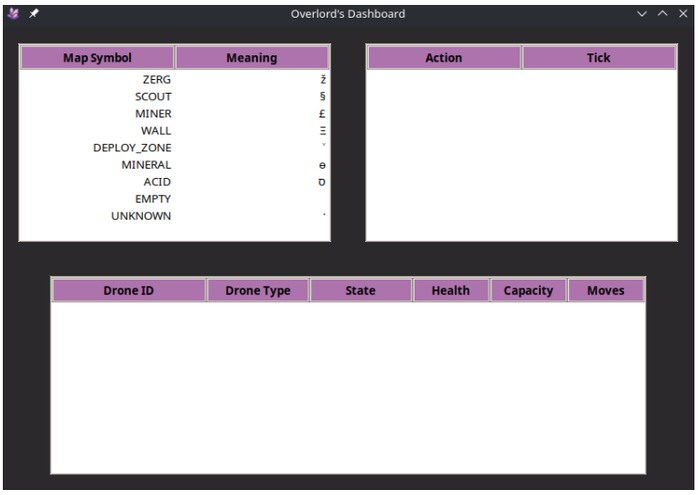

Figure 1: T3 Output

TC4: Add a map

prerequisites: User's code must have previously initialized a dashboard.

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop
from mining.GUI.dashboard import Dashboard
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.icon import Icon
from mining.utils.map import Map
from mining.GUI.dashboard import Dashboard
root = Tk()
example = Dashboard(root)
cocopebbles = Map(0.0)
example.create_map_gui(cocopebbles)
mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user as well as one map that will be filled with unknown tiles.

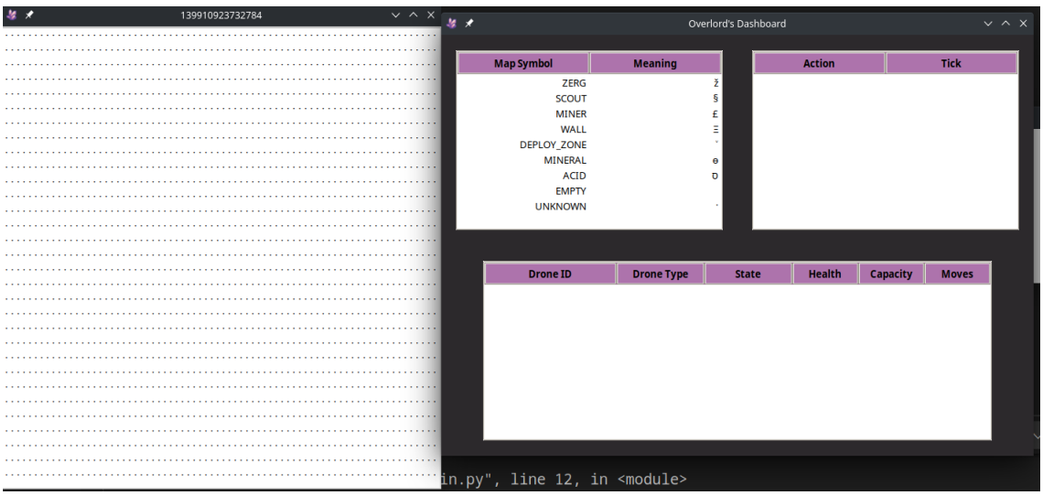

Figure 2: T4 Output

TC5: Add a drone to the dashboard

prerequisites: User's code must have previously initialized a dashboard.

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop
from mining.GUI.dashboard import Dashboard
from mining.zerg_units.drones.scout import ScoutDrone
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.icon import Icon
root = Tk()

example = Dashboard(root)

new_overlord = Overlord(5, 10, example)

new_drone = ScoutDrone(new_overlord)
example.add_drone_to_tree(new_drone)

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user, and will fill the drone table with at least one drone

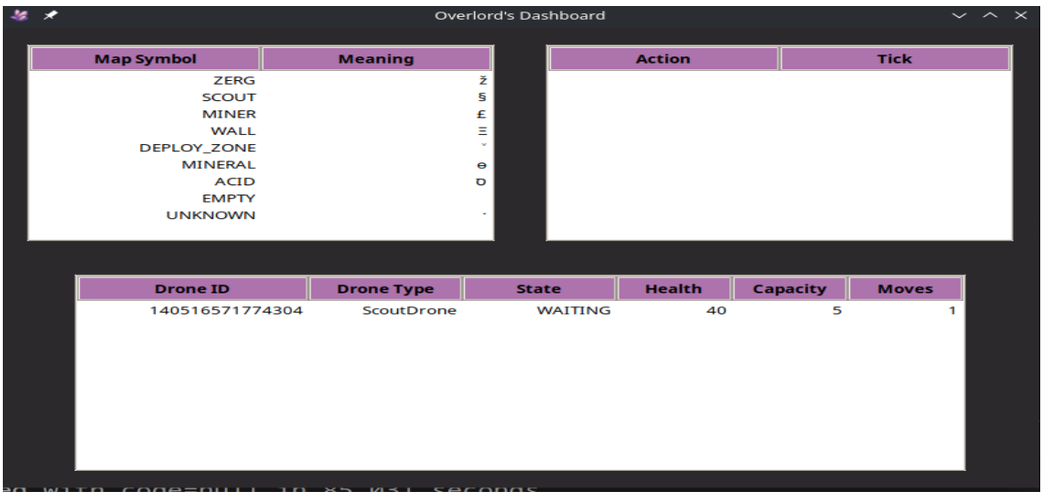

Figure 3: T5 Output

TC6: Add a action to the action table

prerequisites: User's code must have previously initialized a dashboard.

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop
from mining.GUI.dashboard import Dashboard
from mining.zerg_units.drones.scout import ScoutDrone
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.icon import Icon
root = Tk()

example = Dashboard(root)

example.insert_action("Action", "Tick")

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user, and will fill the action table with at least one action and tick.

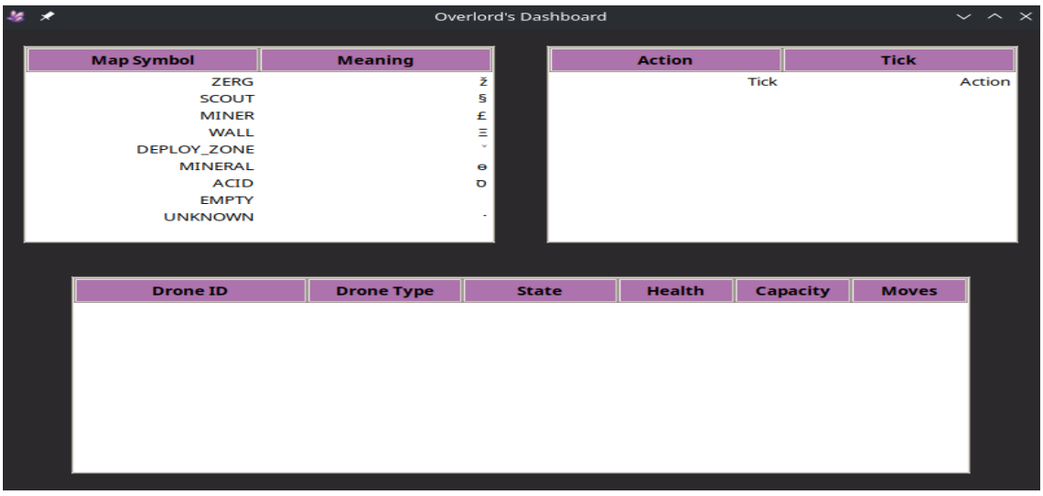

Figure 4: T6 Output

TC7: Reading a map into the GUI

prerequisites: User's code must have previously initialized a dashboard. The user must also have a map object already made. We accomplished this by using a function to read a file into a map

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop

from mining.GUI.dashboard import Dashboard
from mining.utils.map import Map
from mining.utils.tile import Tile
from mining.zerg_units.drones.scout import ScoutDrone
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.coordinate import Coordinate
from mining.utils.icon import Icon



def file_read(file_name):
    """
    Reads a file into a map. This function is just used
    for testing purposes and will not be apart of the main product.
    Argument:
        file_name (string): Name of the file that will be read.
    """
    new_map = Map(0.0)
    Tile_dict = {}
    fp = open(file_name, "r")
    linecounter = 1
    for line in fp:
        lettercounter = 0
        for letter in line:
            if letter == '\n':
                letter = ' '
            elif letter.isnumeric():
                letter = '*'
            Coordinate = (lettercounter, linecounter)
            tile = Tile(Coordinate, Icon(letter))
            Tile_dict[Coordinate] = tile
            lettercounter += 1
        linecounter += 1
    new_map._stored_tiles_ = Tile_dict
    return new_map

root = Tk()
new_map = file_read("maps/map04.txt")

example = Dashboard(root)

example.create_map_gui(new_map)
example.update_maps([])

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user, and will fill the map window with the map that was read.

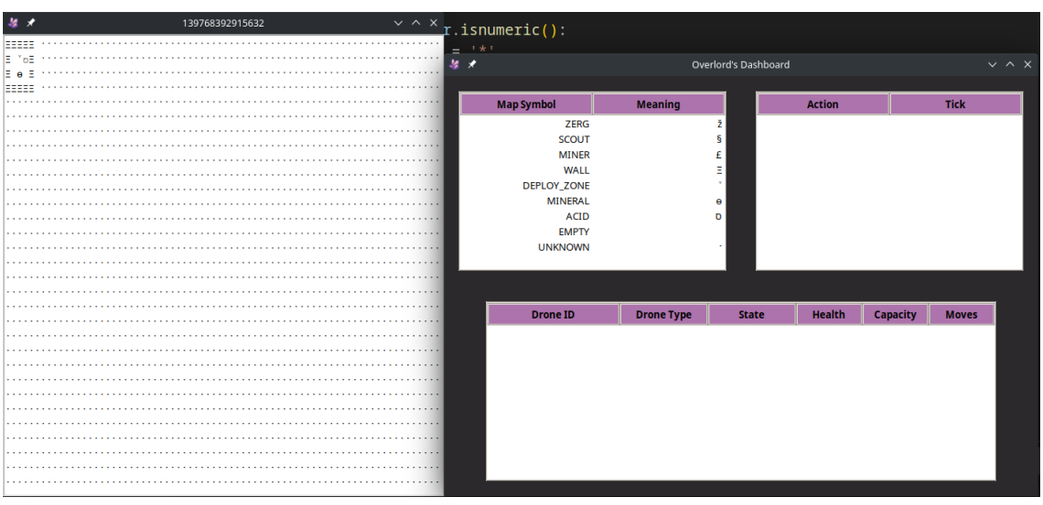

Figure 5: T7 Output

TC8: Reading multiple maps into the GUI

prerequisites: User's code must have previously initialized a dashboard. The user must also have map objects already made. We accomplished this by using a function to read a file into a map

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop

from mining.GUI.dashboard import Dashboard
from mining.utils.map import Map
from mining.utils.tile import Tile
from mining.zerg_units.drones.scout import ScoutDrone
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.coordinate import Coordinate
from mining.utils.icon import Icon



def file_read(file_name):
    """
    Reads a file into a map. This function is just used
    for testing purposes and will not be apart of the main product.
    Argument:
        file_name (string): Name of the file that will be read.
    """
    new_map = Map(0.0)
    Tile_dict = {}
    fp = open(file_name, "r")
    linecounter = 1
    for line in fp:
        lettercounter = 0
        for letter in line:
            if letter == '\n':
                letter = ' '
            elif letter.isnumeric():
                letter = '*'
            Coordinate = (lettercounter, linecounter)
            tile = Tile(Coordinate, Icon(letter))
            Tile_dict[Coordinate] = tile
            lettercounter += 1
        linecounter += 1
    new_map._stored_tiles_ = Tile_dict
    return new_map

root = Tk()
new_map = file_read("maps/map04.txt")
new_map2 = file_read("maps/map03.txt")

example = Dashboard(root)

example.create_map_gui(new_map)
example.create_map_gui(new_map2)
example.update_maps([])

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user, and will fill both map windows with the maps that were read.

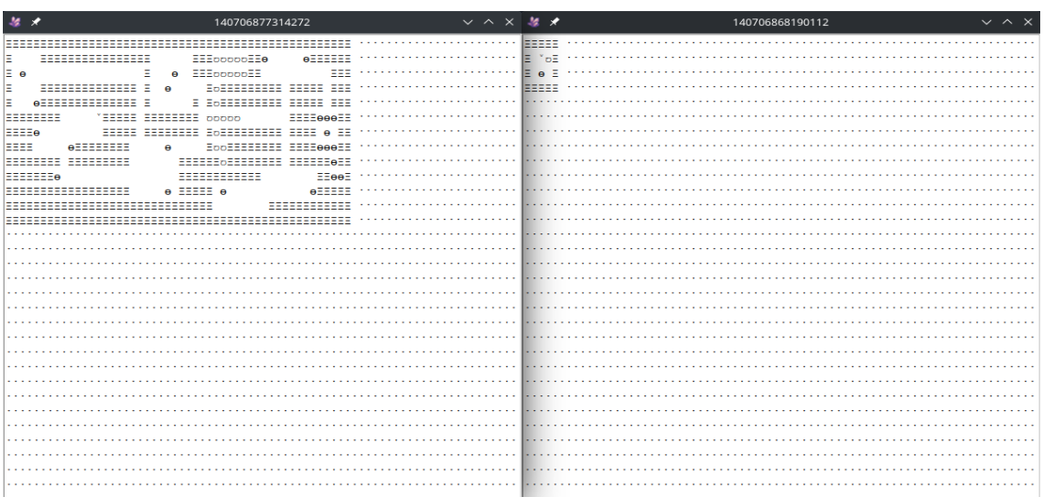

Figure 6: T8 Output

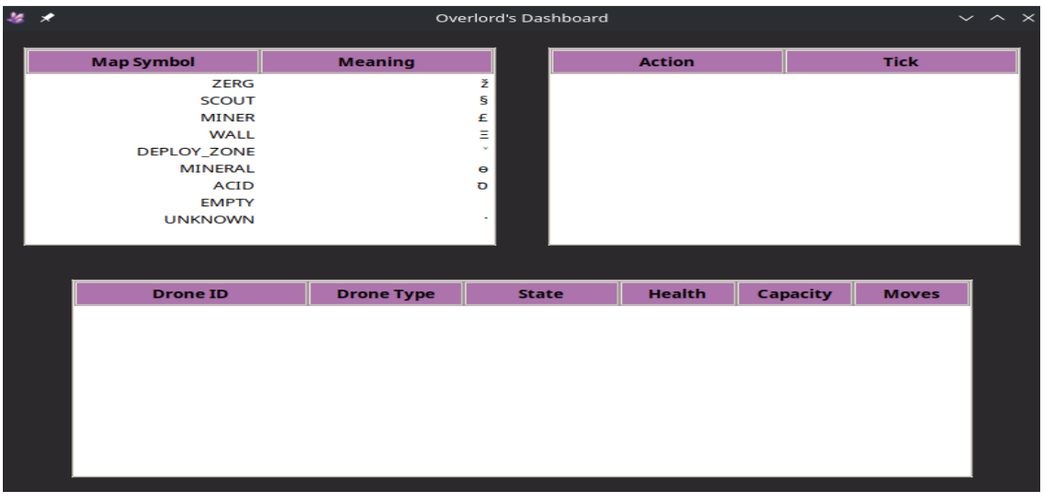

Figure 7: T8 Output

TC9: Reading four maps into the GUI

prerequisites: User's code must have previously initialized a dashboard. The user must also have map objects already made. We accomplished this by using a function to read a file into a map

```
#!/usr/bin/env python3
"""Test main."""
from tkinter import Tk, mainloop

from mining.GUI.dashboard import Dashboard
from mining.utils.map import Map
from mining.utils.tile import Tile
from mining.zerg_units.drones.scout import ScoutDrone
from mining.zerg_units.zerg import Zerg
from mining.zerg_units.overlord import Overlord
from mining.utils.coordinate import Coordinate
from mining.utils.icon import Icon



def file_read(file_name):
    """
    Reads a file into a map. This function is just used
    for testing purposes and will not be apart of the main product.
    Argument:
        file_name (string): Name of the file that will be read.
    """
    new_map = Map(0.0)
    Tile_dict = {}
    fp = open(file_name, "r")
    linecounter = 1
    for line in fp:
        lettercounter = 0
        for letter in line:
            if letter == '\n':
                letter = ' '
            elif letter.isnumeric():
                letter = '*'
            Coordinate = (lettercounter, linecounter)
            tile = Tile(Coordinate, Icon(letter))
            Tile_dict[Coordinate] = tile
            lettercounter += 1
        linecounter += 1
    new_map._stored_tiles_ = Tile_dict
    return new_map

root = Tk()
new_map = file_read("maps/map04.txt")
new_map2 = file_read("maps/map03.txt")
new_map3 = file_read("maps/map02.txt")
new_map4 = file_read("maps/map01.txt")
example = Dashboard(root)

example.create_map_gui(new_map)
example.create_map_gui(new_map2)
example.create_map_gui(new_map3)
example.create_map_gui(new_map4)
example.update_maps([])

mainloop()
```

Expected: The program will initialize a GUI dashboard that will be present to the user, and will fill

both map windows with the maps that were read.

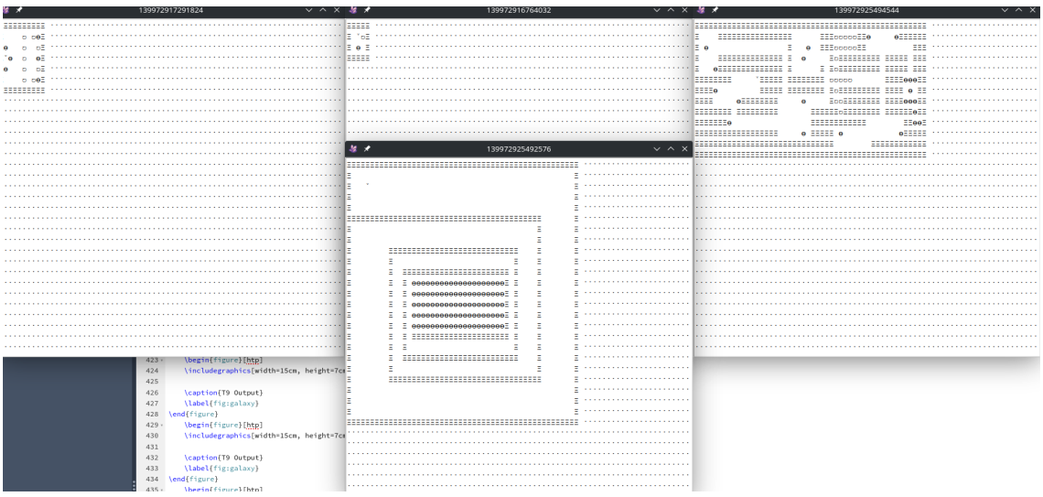

Figure 8: T9 Output

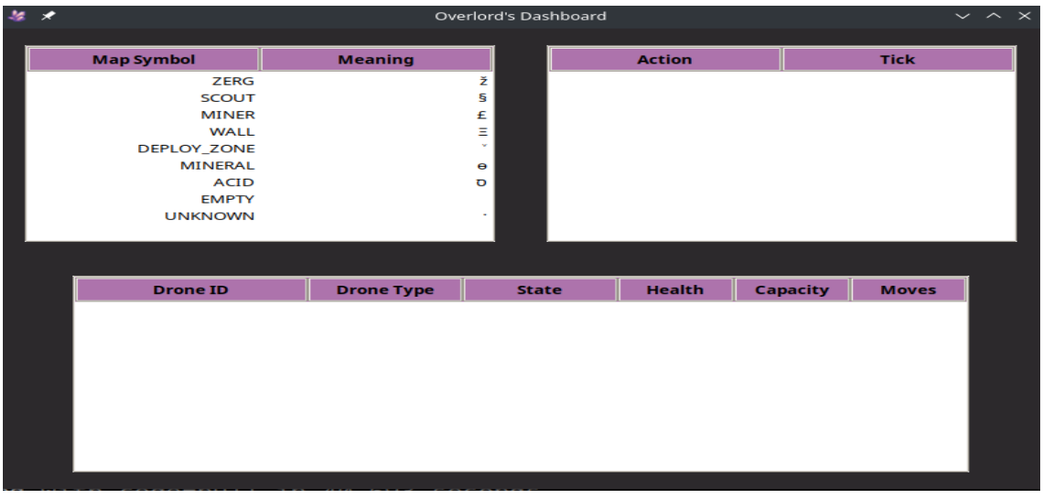

Figure 9: T9 Output

TC10: Launching mining expedition without data.

prerequisites: User must be in mining directory with the begin_mining_expedition.py. Type in the command "python3 begin_mining_expedition.py" into the terminal.

1. *Put the following options in testing menu*
2. *set "Ticks" equal to 100*
3. *set "Refined Minerals" equal to 100*
4. *set "Refresh Delay" equal to 0.1*
5. *Click the start button*
   Expected: The user will get an Error about the program getting no data.