Contents

1 Basic Test Results

```
Tue 17 Jan 2023 23:41:14 IST
Tue 17 Jan 2023 23:41:14 IST
Archive:  /tmp/bodek.mydalp9k/intro2cs1/ex10r1/mallis/final/submission
  inflating: src/apple.py
 extracting: src/AUTHORS
  inflating: src/board.py
  inflating: src/common.py
  inflating: src/results.py
  inflating: src/snake.py
  inflating: src/snake_game.py
  inflating: src/snake_main.py
  inflating: src/wall.py
12 passed tests out of 12 in test set named 'presubmit'.
result_code    presubmit    12    1
Round 149: Error:
Game state differs from expected.
Last round:       (24, {(34, 5): 'black', (34, 6): 'black', (34, 7): 'black', (34, 8): 'black', (33, 8):
'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8):
'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8):
'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8): 'black', (15, 8):
'black', (14, 8): 'black', (34, 4): 'green', (9, 0): 'green', (36, 14): 'green', (23, 7): 'green', (36, 27):
'green', (36, 17): 'green', (31, 2): 'green', (13, 18): 'blue', (12, 18): 'blue', (11, 18): 'blue', (34, 22):
'blue', (33, 22): 'blue', (32, 22): 'blue', (23, 24): 'blue', (23, 25): 'blue', (23, 26): 'blue', (6, 0):
'blue', (6, 1): 'blue', (6, 2): 'blue', (0, 27): 'blue', (0, 26): 'blue', (0, 25): 'blue'})
Current expected: (28, {(34, 4): 'black', (34, 5): 'black', (34, 6): 'black', (34, 7): 'black', (34, 8):
'black', (33, 8): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8):
'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8):
'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8):
'black', (15, 8): 'black', (14, 8): 'black', (9, 0): 'green', (36, 14): 'green', (23, 7): 'green', (36, 27):
'green', (36, 17): 'green', (36, 10): 'green', (31, 2): 'green', (13, 18): 'blue', (12, 18): 'blue', (11,
18): 'blue', (34, 22): 'blue', (33, 22): 'blue', (32, 22): 'blue', (23, 24): 'blue', (23, 25): 'blue', (23,
26): 'blue', (6, 0): 'blue', (6, 1): 'blue', (6, 2): 'blue', (0, 27): 'blue', (0, 26): 'blue', (0, 25):
'blue'})
Current actual:   (29, {(34, 4): 'black', (34, 5): 'black', (34, 6): 'black', (34, 7): 'black', (34, 8):
'black', (33, 8): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8):
'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8):
'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8):
'black', (15, 8): 'black', (14, 8): 'black', (31, 2): 'green', (9, 0): 'green', (36, 17): 'green', (34, 22):
'blue', (33, 22): 'blue', (32, 22): 'blue', (36, 14): 'green', (6, 0): 'blue', (6, 1): 'blue', (6, 2):
'blue', (0, 27): 'blue', (0, 26): 'blue', (0, 25): 'blue', (13, 18): 'blue', (12, 18): 'blue', (11, 18):
'blue', (23, 7): 'green', (23, 24): 'blue', (23, 25): 'blue', (23, 26): 'blue', (36, 27): 'green', (36, 10):
'green'})
--> BEGIN TEST INFORMATION
Test name: moretests_10
Module tested: game_display
Function call: rungame('jjj',Namespace(width=40, height=30, apples=7, debug=False, walls=5,
rounds=-1),['Right', None, None, None, None, None, None, None, None, None, None, None, None, None, 'Up',
None, None, None, None, 'Right', 'Up', None, None, None, None, None, None, 'Left', None, None, None, None,
None, None, 'Up', 'Left', None, None, None, None, None, None, None, None, None, None, None, None, None, None,
None, None, None, None, None, None, None, None, None, None, 'Down', None, None, None, None, None, None,
'Right', None, None, None, None, None, None, None, None, None, None, None, None, None, None, 'Down', None,
None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, 'Left',
None, None, None, None, None, None, None, None, None, None, 'Up', None, None, None, None, None, 'Right',
None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None,
None, None, None, None, None, None, None, None, 'Down', None, None, None, None, None, 'Left', None, None,
None, None, None, None, None, None, None, None, 'Up', None, None, None, None, 'Left', None, None, None, None,
None, None, None, None, None, None, 'Up', 'Right', None, None, None, None, None, None, None, None, None, None,
None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, 'Up', None, None, None,
None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, 'Left', None, None,
```

```
None, None, 'Down', None, None, None, None, None, None, None, None, None, None, 'Right', 'Down', None, None,
None, None, None, None, None, None, None, None, 'Left', None, None, None, None, None, None, None],Ellipsis)
Expected return value: None
More test options: {}
--> END TEST INFORMATION
**********************************************************************
********************     There is a problem:
********************     The test named 'moretests_10' failed.
**********************************************************************
Game state was wrong
result_code    moretests_10    error    1
Round 30: Error:
Game state differs from expected.
Last round:       (1, {(29, 13): 'black', (30, 13): 'black', (31, 13): 'black', (23, 4): 'green', (23, 1):
'green', (6, 24): 'green', (5, 25): 'green', (5, 22): 'green', (13, 14): 'green', (11, 20): 'green', (25,
25): 'green', (7, 22): 'green', (26, 29): 'green', (36, 12): 'green', (24, 4): 'green', (32, 5): 'green', (0,
10): 'green', (1, 5): 'green', (23, 2): 'green', (3, 11): 'green', (30, 7): 'green', (28, 13): 'green', (32,
28): 'green', (2, 21): 'blue', (3, 21): 'blue', (4, 21): 'blue', (2, 5): 'blue', (3, 5): 'blue', (4, 5):
'blue', (22, 8): 'blue', (22, 9): 'blue', (22, 10): 'blue', (34, 8): 'blue', (33, 8): 'blue', (32, 8):
'blue', (10, 0): 'blue', (9, 0): 'blue', (8, 0): 'blue', (13, 13): 'blue', (12, 13): 'blue', (11, 13):
'blue', (21, 29): 'blue', (21, 28): 'blue', (26, 17): 'blue', (25, 17): 'blue', (24, 17): 'blue', (10, 22):
'blue', (10, 21): 'blue', (10, 20): 'blue', (26, 10): 'blue', (26, 9): 'blue', (26, 8): 'blue', (16, 26):
'blue', (17, 26): 'blue', (18, 26): 'blue', (14, 20): 'blue', (15, 20): 'blue', (16, 20): 'blue', (12, 10):
'blue', (12, 11): 'blue', (12, 12): 'blue', (30, 26): 'blue', (29, 26): 'blue', (28, 26): 'blue', (29, 11):
'blue', (30, 11): 'blue', (31, 11): 'blue', (24, 19): 'blue', (24, 20): 'blue', (24, 21): 'blue'})
Current expected: (2, {(28, 13): 'black', (29, 13): 'black', (30, 13): 'black', (31, 13): 'black', (23, 4):
'green', (23, 1): 'green', (6, 24): 'green', (5, 25): 'green', (5, 22): 'green', (13, 14): 'green', (11, 20):
'green', (25, 25): 'green', (7, 22): 'green', (26, 29): 'green', (36, 12): 'green', (24, 4): 'green', (32,
5): 'green', (0, 10): 'green', (23, 2): 'green', (3, 11): 'green', (30, 7): 'green', (24, 12): 'green', (32,
28): 'green', (1, 21): 'blue', (2, 21): 'blue', (3, 21): 'blue', (1, 5): 'blue', (2, 5): 'blue', (3, 5):
'blue', (22, 7): 'blue', (22, 8): 'blue', (22, 9): 'blue', (35, 8): 'blue', (34, 8): 'blue', (33, 8): 'blue',
(11, 0): 'blue', (10, 0): 'blue', (9, 0): 'blue', (14, 13): 'blue', (13, 13): 'blue', (12, 13): 'blue', (21,
29): 'blue', (27, 17): 'blue', (26, 17): 'blue', (25, 17): 'blue', (10, 23): 'blue', (10, 22): 'blue', (10,
21): 'blue', (26, 11): 'blue', (26, 10): 'blue', (26, 9): 'blue', (15, 26): 'blue', (16, 26): 'blue', (17,
26): 'blue', (13, 20): 'blue', (14, 20): 'blue', (15, 20): 'blue', (12, 9): 'blue', (12, 10): 'blue', (12,
11): 'blue', (31, 26): 'blue', (30, 26): 'blue', (29, 26): 'blue', (28, 11): 'blue', (29, 11): 'blue', (30,
11): 'blue', (24, 18): 'blue', (24, 19): 'blue', (24, 20): 'blue'})
Current actual:   (3, {(28, 13): 'black', (29, 13): 'black', (30, 13): 'black', (31, 13): 'black', (35, 8):
'blue', (34, 8): 'blue', (33, 8): 'blue', (30, 7): 'green', (22, 7): 'blue', (22, 8): 'blue', (22, 9):
'blue', (32, 28): 'green', (1, 21): 'blue', (2, 21): 'blue', (3, 21): 'blue', (21, 29): 'blue', (11, 20):
'green', (10, 23): 'blue', (10, 22): 'blue', (10, 21): 'blue', (5, 22): 'green', (13, 20): 'blue', (14, 20):
'blue', (15, 20): 'blue', (26, 29): 'green', (12, 9): 'blue', (12, 10): 'blue', (12, 11): 'blue', (0, 10):
'green', (31, 26): 'blue', (30, 26): 'blue', (29, 26): 'blue', (25, 25): 'green', (5, 25): 'green', (27, 17):
'blue', (26, 17): 'blue', (25, 17): 'blue', (6, 24): 'green', (11, 0): 'blue', (10, 0): 'blue', (9, 0):
'blue', (23, 2): 'green', (1, 5): 'blue', (2, 5): 'blue', (3, 5): 'blue', (13, 14): 'green', (36, 12):
'green', (14, 13): 'blue', (13, 13): 'blue', (12, 13): 'blue', (24, 4): 'green', (26, 11): 'blue', (26, 10):
'blue', (26, 9): 'blue', (7, 22): 'green', (3, 11): 'green', (23, 4): 'green', (28, 11): 'blue', (29, 11):
'blue', (30, 11): 'blue', (23, 1): 'green', (24, 18): 'blue', (24, 19): 'blue', (24, 20): 'blue', (32, 5):
'green', (15, 26): 'blue', (16, 26): 'blue', (17, 26): 'blue', (24, 12): 'green'})
--> BEGIN TEST INFORMATION
Test name: moretests_18
Module tested: game_display
Function call: rungame('rrr',Namespace(width=40, height=30, apples=20, debug=False, walls=16,
rounds=-1),[None, None, None, None, None, None, 'Right', None, None, None, None, None, None, None, None,
None, None, None, 'Down', None, None, None, None, None, None, None, 'Left', None, None, None, None, None,
None, 'Up', None, 'Right', None, None, None, None, None, None, None, 'Down', None, None,
None, None, None, None, None, None, None, 'Left', None, None, None, None, 'Up', None, None, 'Right', None,
None, None, None, 'Up', 'Left', None, None, None, None, None, None, 'Down', None, None, None, None, None,
'Right', None, 'Up', None, None, None, 'Left', None, None, None, None, 'Up', None, None, None, None, None,
None, 'Left', None, None, None, None, None, None, None, None, None, None, None, 'Up', None, 'Right', None,
None, None, None, None, 'Up', None, None, None, None, None, None, None, None, 'Right', None, None, None,
None, None, None, None, None, 'Down', None, None, None, None, None, None, None, 'Left', None, None, None,
None, None, 'Down', None, None, None, None, None, None, None, None, None, None, None, None, None, None, None,
'Right', None, None, None, 'Up', 'Left', None, None, 'Up', None, None, None, None, None, None, None, None,
None, None, None, None, None, None, 'Left', None, None, None, 'Down', None, None, None, 'Right', None, None,
'Down', None, None, None, None, None, None, 'Right', None, None, None, None, None, None, None, 'Up', None, None, None,
None, None, None, None, None, None, None, None, None, None, None, 'Right', None, None, None, 'Up', None, None,
None, None, None, None, None, 'Left', None, None, None, None, None, None, None, 'Up', 'Right', None, None, None, None,
```

```
None, 'Down', None, None, None, None, None, None, None, 'Right', None, None, None, 'Up', None, 'Right', None,
None],Ellipsis)
Expected return value: None
More test options: {}
--> END TEST INFORMATION
**********************************************************************
********************     There is a problem:
********************     The test named 'moretests_18' failed.
**********************************************************************
Game state was wrong
result_code    moretests_18    error    1
24 passed tests out of 26 in test set named 'moretests'.
result_code    moretests    24    1
Listing AUTHORS...
dor_kaiser,mallis:AUTHORSFULL
TESTING COMPLETED
```

2 AUTHORS

```
1 dor_kaiser,mallis
```

3 apple.py

```python
#####################################################################
# FILE : apple.py
# WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
# EXERCISE : intro2cs1 ex10 2023
# DESCRIPTION: Implements the Apple for the Snake Game
# STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
# WEB PAGES I USED: N/A
# NOTES: N/A
#####################################################################

from common import BaseGameObject, Coordinate

class Apple(BaseGameObject):
    """
    Represents an Apple object in the Snake Game.
    This class is used only for matters of interaction
        and it implements no special functionality.
    """

    def __init__(self, coordinate: Coordinate) -> None:
        """
        Creates a new apple in the given coordinate.
        Yes, it's a single coordinate, no oversized apples in this realm.
        """
        super().__init__([coordinate], "green")

    def movement_requirements(self):
        """
        Filler function because the interface must have this implemented.
        """
        return None

    def move(self) -> bool:
        """
        Filler function because the interface must have this implemented.
        """
        return True
```

4 board.py

```
1  ######################################################################
2  # FILE : board.py
3  # WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
4  # EXERCISE : intro2cs1 ex10 2023
5  # DESCRIPTION: Implements the game board for the Snake Game
6  # STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
7  # WEB PAGES I USED: N/A
8  # NOTES: N/A
9  ######################################################################
10
11 from typing import List, Callable, Optional
12 from game_display import GameDisplay
13 from common import Coordinate, BaseGameObject, is_in_boundries, draw_coordinates
14
15 # Signature for the interaction callback, first game object is the source and
16 #   the second game object is the destination
17 InteractionCallback = Callable[[BaseGameObject, BaseGameObject], None]
18 # Signature for the out of bounds callback, called upon interaction of an object
19 #   with the Board's boundries. The boolean parameter is a flag for whether the whole
20 #   object went outside the boundries
21 OutOfBoundsCallback = Callable[[BaseGameObject, bool], bool]
22
23 class Board(object):
24     """
25     Represents the Game Board for Snake.
26     The Board houses generic game objects and is responsible for their state
27     and interaction with the board.
28     """
29
30     def __init__(self, dimensions: Coordinate) -> None:
31         """
32         Creates a new board with the given rows X columns dimensions.
33         """
34         self._dimensions: Coordinate = dimensions
35         self._game_objects: List[BaseGameObject] = []
36
37     def draw_board(self, gui: GameDisplay) -> None:
38         """
39         Draws the board to the given Game Display.
40         The board assumes the Game Display is of the same dimensions
41         as the Board itself.
42         """
43         priority_draw = []
44         priority_color = "blue"
45         for game_object in self._game_objects:
46             if priority_color == game_object.get_object_color:
47                 priority_draw.append(game_object.get_coordinates())
48             else:
49                 draw_coordinates(gui, self._dimensions, game_object.get_coordinates(), game_object.get_object_color())
50
51         for coords in priority_draw:
52             draw_coordinates(gui, self._dimensions, coords, priority_color)
53
54     def add_game_object(self, game_object: BaseGameObject) -> bool:
55         """
56         Adds the game obejct to the board
57         If the object shares coordinates with another object already placed
58         on the board, the method will fail and return False, True otherwise.
59         """
```

```
60          # Making sure the object doesn't overlap with any other object first
61          for coordinate in game_object.get_coordinates():
62              if self._get_object_at_coordinate(coordinate) is not None:
63                  return False
64
65          self._game_objects.append(game_object)
66          return True
67
68      def remove_game_object(self, game_object: BaseGameObject) -> None:
69          """
70          Removing the game object from the board, if it exists
71          """
72          # If the object is not on the board, don't even try to remove it
73          if game_object in self._game_objects:
74              self._game_objects.remove(game_object)
75
76      def move_game_objects(self,
77                            interaction_callback: InteractionCallback,
78                            out_of_bounds_callback: OutOfBoundsCallback) -> bool:
79          """
80          Moving all game objects one after the other, then checks if any
81          object interacts with another, if so, interaction_callback is called.
82          If an object interacts with the boundries of the border,
83          out_of_bounds_callback is called.
84          """
85          interactions = {}
86          out_of_bounds = []
87
88          # Iterating on all objects and moving them
89          for game_object in self._game_objects:
90              requirement = game_object.movement_requirements()
91              # If the game object can move and the requirements are satisfied
92              if requirement is not None and self._is_in_boundries(requirement):
93                  destination_obj =  self._get_object_at_coordinate(requirement)
94
95                  # If there is a game object at the destination,
96                  #   add to possible interactions, it will be checked later if the
97                  #   interaction is still relevant after all objects moved
98                  if game_object.move() and destination_obj is not None:
99                      #interactions.append((game_object, destination_obj))
100                     interactions[(requirement.column, requirement.row)] = (game_object, destination_obj)
101
102             # If the game object is moving out of bounds, let the caller know
103             elif requirement is not None and not self._is_in_boundries(requirement):
104                 game_object.move()
105                 out_of_bounds.append(game_object)
106
107         # Executing interaction between the objects
108         for coordinate, objs in interactions.items():
109             src_obj = objs[0]
110             dst_obj = objs[1]
111             # Making sure after the movement that the objects still interact with eachother
112             if Coordinate.from_legacy_coordinate(coordinate) in dst_obj.get_coordinates():
113                 # If the destination and source objects are the same,
114                 #   then we check that it actually hit itself (since it
115                 #   may have just moved in and out of the cell this round)
116                 # If it is not the same object, then interact normally
117                 if (src_obj is dst_obj and self._is_hitting_itself(src_obj)) or \
118                         (src_obj is not dst_obj):
119                     interaction_callback(src_obj, dst_obj)
120
121         # Executing interaction between objects to the board boundries
122         for object1 in out_of_bounds:
123             out_of_bounds_callback(object1, self._is_off_board(object1.get_coordinates()))
124
125     def _is_hitting_itself(self, object1: BaseGameObject):
126         """
127         Checking if an object is hitting itself
```

```
128             Receives the object which should be checked and returns True if so.
129             """
130             coordinates = object1.get_coordinates()
131             # If the object has no coordinates, we shouldn't check anything
132             if 0 == len(coordinates):
133                 return False
134             # If the object has a coordinate twice, then it hit itself
135             if 1 != coordinates.count(coordinates[0]):
136                 return True
137 
138             return False
139 
140         def _is_interacting(self, coordinate, object1: BaseGameObject, object2: BaseGameObject) -> bool:
141             """
142             Returns whether the two objects interact with each other
143             (e.g. share some coordinate(s) at the given moment)
144             """
145             for coordinate in object1.get_coordinates():
146                 if coordinate in object2.get_coordinates():
147                     return True
148             return False
149 
150         def _get_object_at_coordinate(self, coordinate: Coordinate) -> Optional[BaseGameObject]:
151             """
152             Returns the object residing in the requested coordinate.
153             If no object exists at the specified coordinate None is returned.
154             """
155             for game_object in self._game_objects:
156                 if coordinate in game_object.get_coordinates():
157                     return game_object
158 
159             return None # No object at the specified coordinate
160 
161         def _is_off_board(self, coordinates: List[Coordinate]) -> bool:
162             """
163             Checks if the given coordinates are completely off the game board
164             (e.g. none of the coordinates are within the board's boundries)
165             Returns True if all are off board, False if at least one is on board.
166             """
167             for coordinate in coordinates:
168                 # Checking if at least one coordinate is on the board
169                 if self._is_in_boundries(coordinate):
170                     return False
171             return True
172 
173         def _is_in_boundries(self, coordinate: Coordinate) -> bool:
174             """
175             Checks if the given coordinate is within the board's boundries.
176             Returns True if exists, False otherwise.
177             """
178             return is_in_boundries(self._dimensions.row, self._dimensions.column, coordinate)
```

5 common.py

```python
##################################################################
# FILE : common.py
# WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
# EXERCISE : intro2cs1 ex10 2023
# DESCRIPTION: Common functionality for use around the codebase
# STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
# WEB PAGES I USED: N/A
# NOTES: N/A
##################################################################

from typing import List, Tuple
from game_display import GameDisplay

class Direction(object):
    """
    Used as an Enum for Directions
    """
    LEFT = "Left"
    RIGHT = "Right"
    UP = "Up"
    DOWN = "Down"

    Directions = [LEFT, RIGHT, UP, DOWN]

class Coordinate(object):
    """
    Represents a coordinate in a Two-Dimensional Coordinate System
    """

    def __init__(self, row: int, column: int) -> None:
        """
        Initializes the coordinate with the given Y (row) and X (column) values
        """
        # Attributes are intentionally public, to refrain from getter/setter functions
        self.row = row
        self.column = column

    @staticmethod
    def from_legacy_coordinate(legacy_coordinate: Tuple[int, int]):
        """
        Creating a Coordinate from a legacy (X, Y) Tuple Coordinate
        """
        return Coordinate(legacy_coordinate[1], legacy_coordinate[0])

    def __str__(self) -> str:
        """
        Prints the coordinates in (row, column) format
        """
        return "({row}, {column})".format(row=self.row, column=self.column)

    def __repr__(self) -> str:
        """
        Returns the coordinates in a string format (similar to __str__)
        """
        return self.__str__()

    def __eq__(self, other: object) -> bool:
        """
        Compares one coordinate to another.
```

```
60          A coordinate is equal if both the row and column coordinates are exact.
61          """
62          if (other.row == self.row) and (other.column == self.column):
63              return True
64          return False
65
66  class BaseGameObject(object):
67      """
68      Represents a basic game object for the Snake Game.
69      All game objects set on the board should inherit from this class
70      and implement its functions
71      """
72
73      def __init__(self, coordinates: List[Coordinate], color: str) -> None:
74          """
75          Initializes the object.
76
77          :param coordinates: List of coordinates occupied
78                              by the object on initialization.
79          :param color: The color of the object when shown graphically.
80          """
81          self._coordinates = coordinates
82          self._color = color
83
84      def __str__(self) -> str:
85          """
86          Prints the game object's Coordinates
87          Primarily used for debugging matters.
88          """
89          return str(self._coordinates)
90
91      def __len__(self) -> int:
92          """
93          Returns the length of the object.
94          The length is how many cells the object occupies
95          in a two dimensional coordinate system.
96
97          Builtin len() function should be used.
98          """
99          return len(self._coordinates)
100
101     def get_object_color(self):
102         """
103         Returns the color of the object.
104         """
105         return self._color
106
107     def get_coordinates(self) -> List[Coordinate]:
108         """
109         Returns the coordinates the object occupies.
110         """
111         return self._coordinates
112
113     def movement_requirements(self) -> Coordinate:
114         """
115         Relevant mostly for moving game objects.
116         Should specify the coordinate the game object
117         requires to be empty before moving.
118
119         Calling this directly will throw
120         NotImplementedError Exception.
121         """
122         raise NotImplementedError
123
124     def move(self) -> bool:
125         """
126         Relevant mostly for moving game objects.
127         Should move the object in any way seem fit.
```

```
128
129             Calling this directly will throw
130             NotImplementedError Exception.
131             """
132             raise NotImplementedError
133
134     class BaseDynamicGameObject(BaseGameObject):
135         """
136         Represents a dynamic game object, which is on the move during the game.
137         This is an addition over the regular BaseGameObject in a manner which
138         allows easier interface for more complicated moving objects.
139         """
140
141         def __init__(self, starting_direction, coordinates: List[Coordinate], color: str) -> None:
142             """
143             Initializes the object with the given starting direction,
144             coordinates occupied by the object, and the color of the object on display.
145             """
146             super().__init__(coordinates, color)
147
148             self._current_direction = starting_direction
149
150         def movement_requirements(self) -> Coordinate:
151             """
152             Returns the requirements for the given direction regardless of
153             whether it is valid or invalid (e.g. if the current direction is
154             left and this function is given right)
155             """
156             head = self._coordinates[0] # The head element
157             if Direction.LEFT == self._current_direction:
158                 return Coordinate(head.row, head.column-1)
159             elif Direction.RIGHT == self._current_direction:
160                 return Coordinate(head.row, head.column+1)
161             elif Direction.UP == self._current_direction:
162                 return Coordinate(head.row+1, head.column)
163             else: # Direction is Down
164                 return Coordinate(head.row-1, head.column)
165
166     def is_in_boundries(height: int, width: int, coordinate: Coordinate) -> bool:
167         """
168         Checks if a coordinate is within the given height X width dimensions.
169         """
170         # Checking the row coordinate is within the boundries
171         if coordinate.row >= height or coordinate.row < 0:
172             return False
173
174         # Checking the column coordinate is within the boundries
175         if coordinate.column >= width or coordinate.column < 0:
176             return False
177
178         return True
179
180     def draw_coordinates(
181         gui: GameDisplay, dimensions: Coordinate, coordinates: List[Coordinate], color) -> None:
182         """
183         Draws a list of coordinates on a game display, with the given color.
184         """
185         for coordinate in coordinates:
186             # Only if the coordinate is within the display board draw it
187             if is_in_boundries(dimensions.row, dimensions.column, coordinate):
188                 gui.draw_cell(coordinate.column, coordinate.row, color)
```

6 results.py

```
a = [(0, {(20, 15): 'black', (20, 14): 'black', (20, 13): 'black', (28, 13):
'green', (16, 25): 'blue', (16, 24): 'blue', (16, 23): 'blue'}), (0, {(20, 16): 'black', (20, 15): 'black',
(20, 14): 'black', (21, 14): 'green', (28, 13): 'green', (16, 25): 'blue', (16, 24): 'blue', (16, 23):
'blue', (7, 27): 'blue', (6, 27): 'blue', (5, 27): 'blue'}), (0, {(20, 17): 'black', (20, 16): 'black', (20,
15): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (16, 26): 'blue', (16, 25): 'blue',
(16, 24): 'blue', (8, 27): 'blue', (7, 27): 'blue', (6, 27): 'blue'}), (0, {(20, 18): 'black', (20, 17):
'black', (20, 16): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (16, 26): 'blue', (16,
25): 'blue', (16, 24): 'blue', (8, 27): 'blue', (7, 27): 'blue', (6, 27): 'blue'}), (0, {(20, 19): 'black',
(20, 18): 'black', (20, 17): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (16, 27):
'blue', (16, 26): 'blue', (16, 25): 'blue', (9, 27): 'blue', (8, 27): 'blue', (7, 27): 'blue'}), (0, {(20,
20): 'black', (20, 19): 'black', (20, 18): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green',
(16, 27): 'blue', (16, 26): 'blue', (16, 25): 'blue', (9, 27): 'blue', (8, 27): 'blue', (7, 27): 'blue'}),
(0, {(20, 21): 'black', (20, 20): 'black', (20, 19): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3):
'green', (16, 28): 'blue', (16, 27): 'blue', (16, 26): 'blue', (10, 27): 'blue', (9, 27): 'blue', (8, 27):

'blue'}), (0, {(20, 22): 'black', (20, 21): 'black', (20, 20): 'black', (21, 14): 'green', (28, 13): 'green',
(12, 3): 'green', (16, 28): 'blue', (16, 27): 'blue', (16, 26): 'blue', (10, 27): 'blue', (9, 27): 'blue',
(8, 27): 'blue'}), (0, {(20, 23): 'black', (20, 22): 'black', (20, 21): 'black', (21, 14): 'green', (28, 13):
'green', (12, 3): 'green', (16, 29): 'blue', (16, 28): 'blue', (16, 27): 'blue', (11, 27): 'blue', (10, 27):
'blue', (9, 27): 'blue'}), (0, {(21, 23): 'black', (20, 23): 'black', (20, 22): 'black', (21, 14): 'green',
(28, 13): 'green', (12, 3): 'green', (16, 29): 'blue', (16, 28): 'blue', (16, 27): 'blue', (11, 27): 'blue',
(10, 27): 'blue', (9, 27): 'blue'}), (0, {(22, 23): 'black', (21, 23): 'black', (20, 23): 'black', (21, 14):
'green', (28, 13): 'green', (12, 3): 'green', (16, 29): 'blue', (16, 28): 'blue', (12, 27): 'blue', (11, 27):
'blue', (10, 27): 'blue'}), (0, {(23, 23): 'black', (22, 23): 'black', (21, 23): 'black', (21, 14): 'green',
(28, 13): 'green', (12, 3): 'green', (16, 29): 'blue', (16, 28): 'blue', (12, 27): 'blue', (11, 27): 'blue',
(10, 27): 'blue'}), (0, {(24, 23): 'black', (23, 23): 'black', (22, 23): 'black', (21, 14): 'green', (28,
13): 'green', (12, 3): 'green', (16, 29): 'blue', (13, 27): 'blue', (12, 27): 'blue', (11, 27): 'blue'}), (0,
{(25, 23): 'black', (24, 23): 'black', (23, 23): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3):
'green', (16, 29): 'blue', (13, 27): 'blue', (12, 27): 'blue', (11, 27): 'blue'}), (0, {(26, 23): 'black',
(25, 23): 'black', (24, 23): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (25, 29):
'blue', (25, 28): 'blue', (14, 27): 'blue', (13, 27): 'blue', (12, 27): 'blue'}), (0, {(27, 23): 'black',
(26, 23): 'black', (25, 23): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (25, 29):
'blue', (25, 28): 'blue', (14, 27): 'blue', (13, 27): 'blue', (12, 27): 'blue'}), (0, {(27, 22): 'black',
(27, 23): 'black', (26, 23): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (25, 29):
'blue', (15, 27): 'blue', (14, 27): 'blue', (13, 27): 'blue'}), (0, {(27, 21): 'black', (27, 22): 'black',
(27, 23): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (25, 29): 'blue', (15, 27):
'blue', (14, 27): 'blue', (13, 27): 'blue'}), (0, {(27, 20): 'black', (27, 21): 'black', (27, 22): 'black',
(21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (16, 27): 'blue', (15, 27): 'blue', (14, 27): 'blue',
(6, 10): 'blue', (7, 10): 'blue', (8, 10): 'blue'}), (0, {(27, 19): 'black', (27, 20): 'black', (27, 21):
'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (16, 27): 'blue', (15, 27): 'blue', (14,
27): 'blue', (6, 10): 'blue', (7, 10): 'blue', (8, 10): 'blue'}), (0, {(27, 18): 'black', (27, 19): 'black',
(27, 20): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (17, 27): 'blue', (16, 27):
'blue', (15, 27): 'blue', (5, 10): 'blue', (6, 10): 'blue', (7, 10): 'blue'}), (0, {(27, 17): 'black', (27,
18): 'black', (27, 19): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (17, 27): 'blue',
(16, 27): 'blue', (15, 27): 'blue', (5, 10): 'blue', (6, 10): 'blue', (7, 10): 'blue'}), (0, {(27, 16):
'black', (27, 17): 'black', (27, 18): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (18,
27): 'blue', (17, 27): 'blue', (16, 27): 'blue', (4, 10): 'blue', (5, 10): 'blue', (6, 10): 'blue'}), (0,
{(27, 15): 'black', (27, 16): 'black', (27, 17): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3):
'green', (18, 27): 'blue', (17, 27): 'blue', (16, 27): 'blue', (4, 10): 'blue', (5, 10): 'blue', (6, 10):
'blue'}), (0, {(27, 14): 'black', (27, 15): 'black', (27, 16): 'black', (21, 14): 'green', (28, 13): 'green',
(12, 3): 'green', (19, 27): 'blue', (18, 27): 'blue', (17, 27): 'blue', (3, 10): 'blue', (4, 10): 'blue', (5,
10): 'blue'}), (0, {(28, 14): 'black', (27, 14): 'black', (27, 15): 'black', (21, 14): 'green', (28, 13):
'green', (12, 3): 'green', (19, 27): 'blue', (18, 27): 'blue', (17, 27): 'blue', (3, 10): 'blue', (4, 10):
'blue', (5, 10): 'blue'}), (0, {(29, 14): 'black', (28, 14): 'black', (27, 14): 'black', (21, 14): 'green',
(28, 13): 'green', (12, 3): 'green', (20, 27): 'blue', (19, 27): 'blue', (18, 27): 'blue', (2, 10): 'blue',
(3, 10): 'blue', (4, 10): 'blue'}), (0, {(29, 13): 'black', (29, 14): 'black', (28, 14): 'black', (21, 14):
'green', (28, 13): 'green', (12, 3): 'green', (20, 27): 'blue', (19, 27): 'blue', (18, 27): 'blue', (2, 10):
'blue', (3, 10): 'blue', (4, 10): 'blue'}), (0, {(29, 12): 'black', (29, 13): 'black', (29, 14): 'black',
(21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (21, 27): 'blue', (20, 27): 'blue', (19, 27): 'blue',
```

```
(1, 10): 'blue', (2, 10): 'blue', (3, 10): 'blue'}), (0, {(29, 11): 'black', (29, 12): 'black', (29, 13):
'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (21, 27): 'blue', (20, 27): 'blue', (19,
27): 'blue', (1, 10): 'blue', (2, 10): 'blue', (3, 10): 'blue'}), (0, {(29, 10): 'black', (29, 11): 'black',
(29, 12): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (22, 27): 'blue', (21, 27):
'blue', (20, 27): 'blue', (0, 10): 'blue', (1, 10): 'blue', (2, 10): 'blue'}), (0, {(29, 9): 'black', (29,
10): 'black', (29, 11): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (22, 27): 'blue',
(21, 27): 'blue', (20, 27): 'blue', (0, 10): 'blue', (1, 10): 'blue', (2, 10): 'blue'}), (0, {(29, 8):
'black', (29, 9): 'black', (29, 10): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (23,
27): 'blue', (22, 27): 'blue', (21, 27): 'blue', (0, 10): 'blue', (1, 10): 'blue'}), (0, {(29, 7): 'black',
(29, 8): 'black', (29, 9): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (23, 27): 'blue',
(22, 27): 'blue', (21, 27): 'blue', (0, 10): 'blue', (1, 10): 'blue'}), (0, {(28, 7): 'black', (29, 7):
'black', (29, 8): 'black', (21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (24, 27): 'blue', (23,
27): 'blue', (22, 27): 'blue', (0, 10): 'blue'}), (0, {(28, 8): 'black', (28, 7): 'black', (29, 7): 'black',
(21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (24, 27): 'blue', (23, 27): 'blue', (22, 27): 'blue',
(0, 10): 'blue'}), (0, {(28, 9): 'black', (28, 8): 'black', (28, 7): 'black', (21, 14): 'green', (28, 13):
'green', (12, 3): 'green', (25, 27): 'blue', (24, 27): 'blue', (23, 27): 'blue', (5, 12): 'blue', (4, 12):
'blue', (3, 12): 'blue'}), (0, {(28, 10): 'black', (28, 9): 'black', (28, 8): 'black', (21, 14): 'green',
(28, 13): 'green', (12, 3): 'green', (25, 27): 'blue', (24, 27): 'blue', (23, 27): 'blue', (5, 12): 'blue',
(4, 12): 'blue', (3, 12): 'blue'}), (0, {(28, 11): 'black', (28, 10): 'black', (28, 9): 'black', (21, 14):
'green', (28, 13): 'green', (12, 3): 'green', (26, 27): 'blue', (25, 27): 'blue', (24, 27): 'blue', (6, 12):
'blue', (5, 12): 'blue', (4, 12): 'blue'}), (0, {(28, 12): 'black', (28, 11): 'black', (28, 10): 'black',
(21, 14): 'green', (28, 13): 'green', (12, 3): 'green', (26, 27): 'blue', (25, 27): 'blue', (24, 27): 'blue',
(6, 12): 'blue', (5, 12): 'blue', (4, 12): 'blue'}), (1, {(28, 13): 'black', (28, 12): 'black', (28, 11):
'black', (21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (27, 27): 'blue', (26, 27): 'blue', (25,

27): 'blue', (7, 12): 'blue', (6, 12): 'blue', (5, 12): 'blue'}), (1, {(27, 13): 'black', (28, 13): 'black',
(28, 12): 'black', (28, 11): 'black', (21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (27, 27):
'blue', (26, 27): 'blue', (25, 27): 'blue', (7, 12): 'blue', (6, 12): 'blue', (5, 12): 'blue'}), (1, {(26,
13): 'black', (27, 13): 'black', (28, 13): 'black', (28, 12): 'black', (28, 11): 'black', (21, 14): 'green',
(16, 28): 'green', (12, 3): 'green', (28, 27): 'blue', (27, 27): 'blue', (26, 27): 'blue', (8, 12): 'blue',
(7, 12): 'blue', (6, 12): 'blue'}), (1, {(25, 13): 'black', (26, 13): 'black', (27, 13): 'black', (28, 13):
'black', (28, 12): 'black', (28, 11): 'black', (21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (28,
27): 'blue', (27, 27): 'blue', (26, 27): 'blue', (8, 12): 'blue', (7, 12): 'blue', (6, 12): 'blue'}), (1,
{(24, 13): 'black', (25, 13): 'black', (26, 13): 'black', (27, 13): 'black', (28, 13): 'black', (28, 12):
'black', (21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (29, 27): 'blue', (28, 27): 'blue', (27,
27): 'blue', (9, 12): 'blue', (8, 12): 'blue', (7, 12): 'blue'}), (1, {(23, 13): 'black', (24, 13): 'black',
(25, 13): 'black', (26, 13): 'black', (27, 13): 'black', (28, 13): 'black', (21, 14): 'green', (16, 28):
'green', (12, 3): 'green', (29, 27): 'blue', (28, 27): 'blue', (27, 27): 'blue', (9, 12): 'blue', (8, 12):
'blue', (7, 12): 'blue'}), (1, {(22, 13): 'black', (23, 13): 'black', (24, 13): 'black', (25, 13): 'black',
(26, 13): 'black', (27, 13): 'black', (21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (30, 27):
'blue', (29, 27): 'blue', (28, 27): 'blue', (10, 12): 'blue', (9, 12): 'blue', (8, 12): 'blue'}), (1, {(21,
13): 'black', (22, 13): 'black', (23, 13): 'black', (24, 13): 'black', (25, 13): 'black', (26, 13): 'black',
(21, 14): 'green', (16, 28): 'green', (12, 3): 'green', (30, 27): 'blue', (29, 27): 'blue', (28, 27): 'blue',
(10, 12): 'blue', (9, 12): 'blue', (8, 12): 'blue'}), (3, {(21, 14): 'black', (21, 13): 'black', (22, 13):
'black', (23, 13): 'black', (24, 13): 'black', (25, 13): 'black', (12, 3): 'green', (1, 20): 'green', (16,
28): 'green', (31, 27): 'blue', (30, 27): 'blue', (29, 27): 'blue', (11, 12): 'blue', (10, 12): 'blue', (9,
12): 'blue'}), (3, {(21, 15): 'black', (21, 14): 'black', (21, 13): 'black', (22, 13): 'black', (23, 13):
'black', (24, 13): 'black', (25, 13): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (31,
27): 'blue', (30, 27): 'blue', (29, 27): 'blue', (11, 12): 'blue', (10, 12): 'blue', (9, 12): 'blue'}), (3,
{(21, 16): 'black', (21, 15): 'black', (21, 14): 'black', (21, 13): 'black', (22, 13): 'black', (23, 13):
'black', (24, 13): 'black', (25, 13): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (32,
27): 'blue', (31, 27): 'blue', (30, 27): 'blue', (12, 12): 'blue', (11, 12): 'blue', (10, 12): 'blue'}), (3,
{(21, 17): 'black', (21, 16): 'black', (21, 15): 'black', (21, 14): 'black', (21, 13): 'black', (22, 13):
'black', (23, 13): 'black', (24, 13): 'black', (25, 13): 'black', (12, 3): 'green', (1, 20): 'green', (16,
28): 'green', (32, 27): 'blue', (31, 27): 'blue', (30, 27): 'blue', (12, 12): 'blue', (11, 12): 'blue', (10,
12): 'blue'}), (3, {(20, 17): 'black', (21, 17): 'black', (21, 16): 'black', (21, 15): 'black', (21, 14):
'black', (21, 13): 'black', (22, 13): 'black', (23, 13): 'black', (24, 13): 'black', (12, 3): 'green', (1,
20): 'green', (16, 28): 'green', (33, 27): 'blue', (32, 27): 'blue', (31, 27): 'blue', (13, 12): 'blue', (12,
12): 'blue', (11, 12): 'blue'}), (3, {(19, 17): 'black', (20, 17): 'black', (21, 17): 'black', (21, 16):
'black', (21, 15): 'black', (21, 14): 'black', (21, 13): 'black', (22, 13): 'black', (23, 13): 'black', (12,
3): 'green', (1, 20): 'green', (16, 28): 'green', (33, 27): 'blue', (32, 27): 'blue', (31, 27): 'blue', (13,
12): 'blue', (12, 12): 'blue', (11, 12): 'blue'}), (3, {(18, 17): 'black', (19, 17): 'black', (20, 17):
'black', (21, 17): 'black', (21, 16): 'black', (21, 15): 'black', (21, 14): 'black', (21, 13): 'black', (22,
13): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (34, 27): 'blue', (33, 27): 'blue', (32,
27): 'blue', (14, 12): 'blue', (13, 12): 'blue', (12, 12): 'blue'}), (3, {(17, 17): 'black', (18, 17):
'black', (19, 17): 'black', (20, 17): 'black', (21, 17): 'black', (21, 16): 'black', (21, 15): 'black', (21,
14): 'black', (21, 13): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (34, 27): 'blue',
(33, 27): 'blue', (32, 27): 'blue', (14, 12): 'blue', (13, 12): 'blue', (12, 12): 'blue'}), (3, {(16, 17):
```

```
'black', (17, 17): 'black', (18, 17): 'black', (19, 17): 'black', (20, 17): 'black', (21, 17): 'black', (21,
16): 'black', (21, 15): 'black', (21, 14): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green',
(35, 27): 'blue', (34, 27): 'blue', (33, 27): 'blue', (15, 12): 'blue', (14, 12): 'blue', (13, 12): 'blue'}),
(3, {(16, 18): 'black', (16, 17): 'black', (17, 17): 'black', (18, 17): 'black', (19, 17): 'black', (20, 17):
'black', (21, 17): 'black', (21, 16): 'black', (21, 15): 'black', (12, 3): 'green', (1, 20): 'green', (16,
28): 'green', (35, 27): 'blue', (34, 27): 'blue', (33, 27): 'blue', (15, 12): 'blue', (14, 12): 'blue', (13,
12): 'blue'}), (3, {(16, 19): 'black', (16, 18): 'black', (16, 17): 'black', (17, 17): 'black', (18, 17):
'black', (19, 17): 'black', (20, 17): 'black', (21, 17): 'black', (21, 16): 'black', (12, 3): 'green', (1,
20): 'green', (16, 28): 'green', (36, 27): 'blue', (35, 27): 'blue', (34, 27): 'blue', (16, 12): 'blue', (15,
12): 'blue', (14, 12): 'blue'}), (3, {(16, 20): 'black', (16, 19): 'black', (16, 18): 'black', (16, 17):
'black', (17, 17): 'black', (18, 17): 'black', (19, 17): 'black', (20, 17): 'black', (21, 17): 'black', (12,
3): 'green', (1, 20): 'green', (16, 28): 'green', (36, 27): 'blue', (35, 27): 'blue', (34, 27): 'blue', (16,
12): 'blue', (15, 12): 'blue', (14, 12): 'blue'}), (3, {(16, 21): 'black', (16, 20): 'black', (16, 19):
'black', (16, 18): 'black', (16, 17): 'black', (17, 17): 'black', (18, 17): 'black', (19, 17): 'black', (20,
17): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (37, 27): 'blue', (36, 27): 'blue', (35,
27): 'blue', (17, 12): 'blue', (16, 12): 'blue', (15, 12): 'blue'}), (3, {(16, 22): 'black', (16, 21):
'black', (16, 20): 'black', (16, 19): 'black', (16, 18): 'black', (16, 17): 'black', (17, 17): 'black', (18,
17): 'black', (19, 17): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (37, 27): 'blue',
(36, 27): 'blue', (35, 27): 'blue', (17, 12): 'blue', (16, 12): 'blue', (15, 12): 'blue'}), (3, {(16, 23):
'black', (16, 22): 'black', (16, 21): 'black', (16, 20): 'black', (16, 19): 'black', (16, 18): 'black', (16,
17): 'black', (17, 17): 'black', (18, 17): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green',
(38, 27): 'blue', (37, 27): 'blue', (36, 27): 'blue', (18, 12): 'blue', (17, 12): 'blue', (16, 12): 'blue'}),
(3, {(16, 24): 'black', (16, 23): 'black', (16, 22): 'black', (16, 21): 'black', (16, 20): 'black', (16, 19):
'black', (16, 18): 'black', (16, 17): 'black', (17, 17): 'black', (12, 3): 'green', (1, 20): 'green', (16,
28): 'green', (38, 27): 'blue', (37, 27): 'blue', (36, 27): 'blue', (18, 12): 'blue', (17, 12): 'blue', (16,

12): 'blue'}), (3, {(16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (16, 22): 'black', (16, 21):
'black', (16, 20): 'black', (16, 19): 'black', (16, 18): 'black', (16, 17): 'black', (12, 3): 'green', (1,
20): 'green', (16, 28): 'green', (39, 27): 'blue', (38, 27): 'blue', (37, 27): 'blue', (19, 12): 'blue', (18,
12): 'blue', (17, 12): 'blue'}), (3, {(16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16, 23):
'black', (16, 22): 'black', (16, 21): 'black', (16, 20): 'black', (16, 19): 'black', (16, 18): 'black', (12,
3): 'green', (1, 20): 'green', (16, 28): 'green', (39, 27): 'blue', (38, 27): 'blue', (37, 27): 'blue', (19,
12): 'blue', (18, 12): 'blue', (17, 12): 'blue'}), (3, {(16, 27): 'black', (16, 26): 'black', (16, 25):
'black', (16, 24): 'black', (16, 23): 'black', (16, 22): 'black', (16, 21): 'black', (16, 20): 'black', (16,
19): 'black', (12, 3): 'green', (1, 20): 'green', (16, 28): 'green', (39, 27): 'blue', (38, 27): 'blue', (20,
12): 'blue', (19, 12): 'blue', (18, 12): 'blue'}), (6, {(16, 28): 'black', (16, 27): 'black', (16, 26):
'black', (16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (16, 22): 'black', (16, 21): 'black', (16,
20): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green', (39, 27): 'blue', (38, 27): 'blue', (20,
12): 'blue', (19, 12): 'blue', (18, 12): 'blue'}), (6, {(15, 28): 'black', (16, 28): 'black', (16, 27):
'black', (16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (16, 22): 'black', (16,
21): 'black', (16, 20): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green', (39, 27): 'blue', (21,
12): 'blue', (20, 12): 'blue', (19, 12): 'blue'}), (6, {(14, 28): 'black', (15, 28): 'black', (16, 28):
'black', (16, 27): 'black', (16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (16,
22): 'black', (16, 21): 'black', (16, 20): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green',
(39, 27): 'blue', (21, 12): 'blue', (20, 12): 'blue', (19, 12): 'blue'}), (6, {(13, 28): 'black', (14, 28):
'black', (15, 28): 'black', (16, 28): 'black', (16, 27): 'black', (16, 26): 'black', (16, 25): 'black', (16,
24): 'black', (16, 23): 'black', (16, 22): 'black', (16, 21): 'black', (16, 20): 'black', (12, 3): 'green',
(5, 15): 'green', (1, 20): 'green', (12, 7): 'blue', (12, 8): 'blue', (12, 9): 'blue', (22, 12): 'blue', (21,
12): 'blue', (20, 12): 'blue'}), (6, {(12, 28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28):
'black', (16, 28): 'black', (16, 27): 'black', (16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16,
23): 'black', (16, 22): 'black', (16, 21): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green',
(12, 7): 'blue', (12, 8): 'blue', (12, 9): 'blue', (22, 12): 'blue', (21, 12): 'blue', (20, 12): 'blue'}),
(6, {(11, 28): 'black', (12, 28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28): 'black', (16, 28):
'black', (16, 27): 'black', (16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (16,
22): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green', (12, 6): 'blue', (12, 7): 'blue', (12,
8): 'blue', (23, 12): 'blue', (22, 12): 'blue', (21, 12): 'blue'}), (6, {(10, 28): 'black', (11, 28):
'black', (12, 28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28): 'black', (16, 28): 'black', (16,
27): 'black', (16, 26): 'black', (16, 25): 'black', (16, 24): 'black', (16, 23): 'black', (12, 3): 'green',
(5, 15): 'green', (1, 20): 'green', (12, 6): 'blue', (12, 7): 'blue', (12, 8): 'blue', (23, 12): 'blue', (22,
12): 'blue', (21, 12): 'blue'}), (6, {(9, 28): 'black', (10, 28): 'black', (11, 28): 'black', (12, 28):
'black', (13, 28): 'black', (14, 28): 'black', (15, 28): 'black', (16, 28): 'black', (16, 27): 'black', (16,
26): 'black', (16, 25): 'black', (16, 24): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green',
(12, 5): 'blue', (12, 6): 'blue', (12, 7): 'blue', (24, 12): 'blue', (23, 12): 'blue', (22, 12): 'blue'}),
(6, {(8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (11, 28): 'black', (12, 28): 'black', (13, 28):
'black', (14, 28): 'black', (15, 28): 'black', (16, 28): 'black', (16, 27): 'black', (16, 26): 'black', (16,
25): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green', (12, 5): 'blue', (12, 6): 'blue', (12,
7): 'blue', (24, 12): 'blue', (23, 12): 'blue', (22, 12): 'blue'}), (6, {(7, 28): 'black', (8, 28): 'black',
(9, 28): 'black', (10, 28): 'black', (11, 28): 'black', (12, 28): 'black', (13, 28): 'black', (14, 28):
```

```
'black', (15, 28): 'black', (16, 28): 'black', (16, 27): 'black', (16, 26): 'black', (12, 3): 'green', (5,
15): 'green', (1, 20): 'green', (12, 4): 'blue', (12, 5): 'blue', (12, 6): 'blue', (25, 12): 'blue', (24,
12): 'blue', (23, 12): 'blue'}), (6, {(6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9, 28): 'black',
(10, 28): 'black', (11, 28): 'black', (12, 28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28):
'black', (16, 28): 'black', (16, 27): 'black', (12, 3): 'green', (5, 15): 'green', (1, 20): 'green', (12, 4):
'blue', (12, 5): 'blue', (12, 6): 'blue', (25, 12): 'blue', (24, 12): 'blue', (23, 12): 'blue'}), (6, {(5,
28): 'black', (6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (11,
28): 'black', (12, 28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28): 'black', (16, 28): 'black',
(5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12, 3): 'blue', (12, 4): 'blue', (12, 5): 'blue',
(26, 12): 'blue', (25, 12): 'blue', (24, 12): 'blue'}), (6, {(4, 28): 'black', (5, 28): 'black', (6, 28):
'black', (7, 28): 'black', (8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (11, 28): 'black', (12,
28): 'black', (13, 28): 'black', (14, 28): 'black', (15, 28): 'black', (5, 15): 'green', (33, 21): 'green',
(1, 20): 'green', (12, 3): 'blue', (12, 4): 'blue', (12, 5): 'blue', (26, 12): 'blue', (25, 12): 'blue', (24,
12): 'blue'}), (6, {(3, 28): 'black', (4, 28): 'black', (5, 28): 'black', (6, 28): 'black', (7, 28): 'black',
(8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (11, 28): 'black', (12, 28): 'black', (13, 28):
'black', (14, 28): 'black', (5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12, 2): 'blue', (12, 3):
'blue', (12, 4): 'blue', (27, 12): 'blue', (26, 12): 'blue', (25, 12): 'blue'}), (6, {(2, 28): 'black', (3,
28): 'black', (4, 28): 'black', (5, 28): 'black', (6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9,
28): 'black', (10, 28): 'black', (11, 28): 'black', (12, 28): 'black', (13, 28): 'black', (5, 15): 'green',
(33, 21): 'green', (1, 20): 'green', (12, 2): 'blue', (12, 3): 'blue', (12, 4): 'blue', (27, 12): 'blue',
(26, 12): 'blue', (25, 12): 'blue'}), (6, {(2, 27): 'black', (2, 28): 'black', (3, 28): 'black', (4, 28):
'black', (5, 28): 'black', (6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9, 28): 'black', (10, 28):
'black', (11, 28): 'black', (12, 28): 'black', (5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12,
1): 'blue', (12, 2): 'blue', (12, 3): 'blue', (28, 12): 'blue', (27, 12): 'blue', (26, 12): 'blue'}), (6,
{(2, 26): 'black', (2, 27): 'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 28): 'black',
(6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (11, 28): 'black',

(5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12, 1): 'blue', (12, 2): 'blue', (12, 3): 'blue',
(28, 12): 'blue', (27, 12): 'blue', (26, 12): 'blue'}), (6, {(2, 25): 'black', (2, 26): 'black', (2, 27):
'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 28): 'black', (6, 28): 'black', (7, 28):
'black', (8, 28): 'black', (9, 28): 'black', (10, 28): 'black', (5, 15): 'green', (33, 21): 'green', (1, 20):
'green', (12, 0): 'blue', (12, 1): 'blue', (12, 2): 'blue', (29, 12): 'blue', (28, 12): 'blue', (27, 12):
'blue'}), (6, {(2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2, 28): 'black', (3,
28): 'black', (4, 28): 'black', (5, 28): 'black', (6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (9,
28): 'black', (5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12, 0): 'blue', (12, 1): 'blue', (12,
2): 'blue', (29, 12): 'blue', (28, 12): 'blue', (27, 12): 'blue'}), (6, {(2, 23): 'black', (2, 24): 'black',
(2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black',
(5, 28): 'black', (6, 28): 'black', (7, 28): 'black', (8, 28): 'black', (5, 15): 'green', (33, 21): 'green',
(1, 20): 'green', (12, 0): 'blue', (12, 1): 'blue', (30, 12): 'blue', (29, 12): 'blue', (28, 12): 'blue'}),
(6, {(2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27):
'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 28): 'black', (6, 28): 'black', (7, 28):
'black', (5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12, 0): 'blue', (12, 1): 'blue', (30, 12):
'blue', (29, 12): 'blue', (28, 12): 'blue'}), (6, {(2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (2,
24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2, 28): 'black', (3, 28): 'black', (4,
28): 'black', (5, 28): 'black', (6, 28): 'black', (5, 15): 'green', (33, 21): 'green', (1, 20): 'green', (12,
0): 'blue', (31, 12): 'blue', (30, 12): 'blue', (29, 12): 'blue'}), (6, {(2, 20): 'black', (2, 21): 'black',
(2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black',
(2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 28): 'black', (5, 15): 'green', (33, 21): 'green',
(1, 20): 'green', (12, 0): 'blue', (31, 12): 'blue', (30, 12): 'blue', (29, 12): 'blue'}), (9, {(1, 20):
'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25):
'black', (2, 26): 'black', (2, 27): 'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 15):
'green', (38, 15): 'green', (33, 21): 'green', (8, 29): 'blue', (8, 28): 'blue', (8, 27): 'blue', (32, 12):
'blue', (31, 12): 'blue', (30, 12): 'blue'}), (9, {(1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2,
21): 'black', (2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2,
27): 'black', (2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 15): 'green', (38, 15): 'green', (33,
21): 'green', (8, 29): 'blue', (8, 28): 'blue', (8, 27): 'blue', (32, 12): 'blue', (31, 12): 'blue', (30,
12): 'blue'}), (9, {(1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black',
(2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black',
(2, 28): 'black', (3, 28): 'black', (4, 28): 'black', (5, 15): 'green', (38, 15): 'green', (33, 21): 'green',
(8, 29): 'blue', (8, 28): 'blue', (33, 12): 'blue', (32, 12): 'blue', (31, 12): 'blue'}), (9, {(1, 17):
'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22):
'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2, 28):
'black', (3, 28): 'black', (4, 28): 'black', (5, 15): 'green', (38, 15): 'green', (33, 21): 'green', (8, 29):
'blue', (8, 28): 'blue', (33, 12): 'blue', (32, 12): 'blue', (31, 12): 'blue'}), (9, {(1, 16): 'black', (1,
17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2,
22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2,
28): 'black', (3, 28): 'black', (5, 15): 'green', (38, 15): 'green', (33, 21): 'green', (8, 29): 'blue', (34,
12): 'blue', (33, 12): 'blue', (32, 12): 'blue'}), (9, {(2, 16): 'black', (1, 16): 'black', (1, 17): 'black',
```

```
(1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black',
(2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (2, 28): 'black',
(5, 15): 'green', (38, 15): 'green', (33, 21): 'green', (8, 29): 'blue', (34, 12): 'blue', (33, 12): 'blue',
(32, 12): 'blue'}), (9, {(3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black', (1, 18):
'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23):
'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (2, 27): 'black', (5, 15): 'green', (38, 15):
'green', (33, 21): 'green', (1, 24): 'blue', (1, 25): 'blue', (1, 26): 'blue', (35, 12): 'blue', (34, 12):
'blue', (33, 12): 'blue'}), (9, {(4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1,
17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2,
22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (2, 26): 'black', (5, 15): 'green', (38,
15): 'green', (33, 21): 'green', (1, 24): 'blue', (1, 25): 'blue', (1, 26): 'blue', (35, 12): 'blue', (34,
12): 'blue', (33, 12): 'blue'}), (9, {(5, 16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black',
(1, 16): 'black', (1, 17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black',
(2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (2, 25): 'black', (5, 15): 'green',
(38, 15): 'green', (33, 21): 'green', (1, 23): 'blue', (1, 24): 'blue', (1, 25): 'blue', (36, 12): 'blue',
(35, 12): 'blue', (34, 12): 'blue'}), (12, {(5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (3, 16):
'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20):
'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (38, 15):
'green', (33, 21): 'green', (31, 13): 'green', (1, 23): 'blue', (1, 24): 'blue', (1, 25): 'blue', (36, 12):
'blue', (35, 12): 'blue', (34, 12): 'blue'}), (12, {(5, 14): 'black', (5, 15): 'black', (5, 16): 'black', (4,
16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black', (1, 18): 'black', (1,
19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (2,
24): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1, 22): 'blue', (1, 23): 'blue', (1,
24): 'blue', (37, 12): 'blue', (36, 12): 'blue', (35, 12): 'blue'}), (12, {(5, 13): 'black', (5, 14):
'black', (5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16):
'black', (1, 17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21):
'black', (2, 22): 'black', (2, 23): 'black', (2, 24): 'black', (38, 15): 'green', (33, 21): 'green', (31,

13): 'green', (1, 22): 'blue', (1, 23): 'blue', (1, 24): 'blue', (37, 12): 'blue', (36, 12): 'blue', (35,
12): 'blue'}), (12, {(5, 12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black', (5, 16):
'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black', (1, 18):
'black', (1, 19): 'black', (1, 20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23):
'black', (2, 24): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1, 21): 'blue', (1, 22):
'blue', (1, 23): 'blue', (38, 12): 'blue', (37, 12): 'blue', (36, 12): 'blue'}), (12, {(5, 11): 'black', (5,
12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (3,
16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black', (1, 18): 'black', (1, 19): 'black', (1,
20): 'black', (2, 20): 'black', (2, 21): 'black', (2, 22): 'black', (2, 23): 'black', (38, 15): 'green', (33,
21): 'green', (31, 13): 'green', (1, 21): 'blue', (1, 22): 'blue', (1, 23): 'blue', (38, 12): 'blue', (37,
12): 'blue', (36, 12): 'blue'}), (12, {(5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13):
'black', (5, 14): 'black', (5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16):
'black', (1, 16): 'black', (1, 17): 'black', (1, 18): 'black', (1, 19): 'black', (1, 20): 'blue', (2, 20):
'black', (2, 21): 'black', (2, 22): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1,
21): 'blue', (1, 22): 'blue', (39, 12): 'blue', (38, 12): 'blue', (37, 12): 'blue'}), (12, {(5, 9): 'black',
(5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black',
(5, 16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17): 'black',
(1, 18): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1, 20): 'blue', (1, 21): 'blue',
(1, 22): 'blue', (39, 12): 'blue', (38, 12): 'blue', (37, 12): 'blue'}), (12, {(5, 8): 'black', (5, 9):
'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15):
'black', (5, 16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (1, 17):
'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1, 19): 'blue', (1, 20): 'blue', (1, 21):
'blue', (39, 12): 'blue', (38, 12): 'blue'}), (12, {(6, 8): 'black', (5, 8): 'black', (5, 9): 'black', (5,
10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black', (5,
16): 'black', (4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (1, 16): 'black', (38, 15): 'green', (33,
21): 'green', (31, 13): 'green', (1, 19): 'blue', (1, 20): 'blue', (1, 21): 'blue', (39, 12): 'blue', (38,
12): 'blue'}), (12, {(7, 8): 'black', (6, 8): 'black', (5, 8): 'black', (5, 9): 'black', (5, 10): 'black',
(5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black', (5, 16): 'black',
(4, 16): 'black', (3, 16): 'black', (2, 16): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (1, 18): 'blue', (1, 19): 'blue', (1, 20): 'blue', (39, 12): 'blue'}), (12, {(8, 8): 'black', (7,
8): 'black', (6, 8): 'black', (5, 8): 'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12):
'black', (5, 13): 'black', (5, 14): 'black', (5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (3, 16):
'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (1, 18): 'blue', (1, 19): 'blue', (1, 20):
'blue', (39, 12): 'blue'}), (12, {(9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5, 8):
'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14):
'black', (5, 15): 'black', (5, 16): 'black', (4, 16): 'black', (38, 15): 'green', (33, 21): 'green', (31,
13): 'green', (6, 25): 'blue', (5, 25): 'blue', (4, 25): 'blue', (1, 17): 'blue', (1, 18): 'blue', (1, 19):
'blue'}), (12, {(10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5, 8):
'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5, 14):
'black', (5, 15): 'black', (5, 16): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (6,
```

```
25): 'blue', (5, 25): 'blue', (4, 25): 'blue', (1, 17): 'blue', (1, 18): 'blue', (1, 19): 'blue'}), (12,
{(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5,
14): 'black', (5, 15): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (7, 25): 'blue', (6,
25): 'blue', (5, 25): 'blue', (1, 16): 'blue', (1, 17): 'blue', (1, 18): 'blue'}), (12, {(12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (5,
14): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (7, 25): 'blue', (6, 25): 'blue', (5,
25): 'blue', (1, 16): 'blue', (1, 17): 'blue', (1, 18): 'blue'}), (12, {(13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (5, 13): 'black', (38,
15): 'green', (33, 21): 'green', (31, 13): 'green', (8, 25): 'blue', (7, 25): 'blue', (6, 25): 'blue', (1,
15): 'blue', (1, 16): 'blue', (1, 17): 'blue'}), (12, {(14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (5, 12): 'black', (38, 15): 'green', (33,
21): 'green', (31, 13): 'green', (8, 25): 'blue', (7, 25): 'blue', (6, 25): 'blue', (1, 15): 'blue', (1, 16):
'blue', (1, 17): 'blue'}), (12, {(15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black', (11,
8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5, 8):
'black', (5, 9): 'black', (5, 10): 'black', (5, 11): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (9, 25): 'blue', (8, 25): 'blue', (7, 25): 'blue', (1, 14): 'blue', (1, 15): 'blue', (1, 16):
'blue'}), (12, {(16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (5, 10): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (9,
25): 'blue', (8, 25): 'blue', (7, 25): 'blue', (1, 14): 'blue', (1, 15): 'blue', (1, 16): 'blue'}), (12,
{(17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (5, 9): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (10, 25): 'blue', (9,
25): 'blue', (8, 25): 'blue', (1, 13): 'blue', (1, 14): 'blue', (1, 15): 'blue'}), (12, {(18, 8): 'black',

(17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (5,
8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13): 'green', (10, 25): 'blue', (9, 25): 'blue', (8,
25): 'blue', (1, 13): 'blue', (1, 14): 'blue', (1, 15): 'blue'}), (12, {(19, 8): 'black', (18, 8): 'black',
(17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (6, 8): 'black', (38,
15): 'green', (33, 21): 'green', (31, 13): 'green', (11, 25): 'blue', (10, 25): 'blue', (9, 25): 'blue', (1,
12): 'blue', (1, 13): 'blue', (1, 14): 'blue'}), (12, {(20, 8): 'black', (19, 8): 'black', (18, 8): 'black',
(17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black',
(11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (7, 8): 'black', (38, 15): 'green',
(33, 21): 'green', (31, 13): 'green', (11, 25): 'blue', (10, 25): 'blue', (9, 25): 'blue', (1, 12): 'blue',
(1, 13): 'blue', (1, 14): 'blue'}), (12, {(21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8):
'black', (17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12, 8):
'black', (11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (8, 8): 'black', (38, 15): 'green', (33, 21):
'green', (31, 13): 'green', (12, 25): 'blue', (11, 25): 'blue', (10, 25): 'blue', (1, 11): 'blue', (1, 12):
'blue', (1, 13): 'blue'}), (12, {(22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18,
8): 'black', (17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (12,
8): 'black', (11, 8): 'black', (10, 8): 'black', (9, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31,
13): 'green', (12, 25): 'blue', (11, 25): 'blue', (10, 25): 'blue', (1, 11): 'blue', (1, 12): 'blue', (1,
13): 'blue'}), (12, {(23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (19, 8):
'black', (18, 8): 'black', (17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (13, 8):
'black', (12, 8): 'black', (11, 8): 'black', (10, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31,
13): 'green', (13, 25): 'blue', (12, 25): 'blue', (11, 25): 'blue', (1, 10): 'blue', (1, 11): 'blue', (1,
12): 'blue'}), (12, {(24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (20, 8):
'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (14, 8):
'black', (13, 8): 'black', (12, 8): 'black', (11, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31,
13): 'green', (13, 25): 'blue', (12, 25): 'blue', (11, 25): 'blue', (1, 10): 'blue', (1, 11): 'blue', (1,
12): 'blue'}), (12, {(25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8):
'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8): 'black', (15, 8):
'black', (14, 8): 'black', (13, 8): 'black', (12, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31,
13): 'green', (14, 25): 'blue', (13, 25): 'blue', (12, 25): 'blue', (1, 9): 'blue', (1, 10): 'blue', (1, 11):
'blue'}), (12, {(26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black',
(21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (16, 8): 'black',
(15, 8): 'black', (14, 8): 'black', (13, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (14, 25): 'blue', (13, 25): 'blue', (12, 25): 'blue', (1, 9): 'blue', (1, 10): 'blue', (1, 11):
'blue'}), (12, {(27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black',
(22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (17, 8): 'black',
(16, 8): 'black', (15, 8): 'black', (14, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (15, 25): 'blue', (14, 25): 'blue', (13, 25): 'blue', (1, 8): 'blue', (1, 9): 'blue', (1, 10):
```

```
'blue'}), (12, {(28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black',
(23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (18, 8): 'black',
(17, 8): 'black', (16, 8): 'black', (15, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (15, 25): 'blue', (14, 25): 'blue', (13, 25): 'blue', (1, 8): 'blue', (1, 9): 'blue', (1, 10):
'blue'}), (12, {(29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black',
(24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (19, 8): 'black',
(18, 8): 'black', (17, 8): 'black', (16, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (16, 25): 'blue', (15, 25): 'blue', (14, 25): 'blue', (1, 7): 'blue', (1, 8): 'blue', (1, 9):
'blue'}), (12, {(30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black',
(25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (20, 8): 'black',
(19, 8): 'black', (18, 8): 'black', (17, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (16, 25): 'blue', (15, 25): 'blue', (14, 25): 'blue', (1, 7): 'blue', (1, 8): 'blue', (1, 9):
'blue'}), (12, {(31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black',
(26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (21, 8): 'black',
(20, 8): 'black', (19, 8): 'black', (18, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (17, 25): 'blue', (16, 25): 'blue', (15, 25): 'blue', (1, 6): 'blue', (1, 7): 'blue', (1, 8):
'blue'}), (12, {(32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black',
(27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (22, 8): 'black',
(21, 8): 'black', (20, 8): 'black', (19, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (17, 25): 'blue', (16, 25): 'blue', (15, 25): 'blue', (1, 6): 'blue', (1, 7): 'blue', (1, 8):
'blue'}), (12, {(32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black',
(28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (23, 8): 'black',
(22, 8): 'black', (21, 8): 'black', (20, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (18, 25): 'blue', (17, 25): 'blue', (16, 25): 'blue', (1, 5): 'blue', (1, 6): 'blue', (1, 7):
'blue'}), (12, {(32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black',
(29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (24, 8): 'black',
(23, 8): 'black', (22, 8): 'black', (21, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (18, 25): 'blue', (17, 25): 'blue', (16, 25): 'blue', (1, 5): 'blue', (1, 6): 'blue', (1, 7):
'blue'}), (12, {(32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black',

(30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black',
(24, 8): 'black', (23, 8): 'black', (22, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (19, 25): 'blue', (18, 25): 'blue', (17, 25): 'blue', (1, 4): 'blue', (1, 5): 'blue', (1, 6):
'blue'}), (12, {(32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black',
(31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black',
(25, 8): 'black', (24, 8): 'black', (23, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (19, 25): 'blue', (18, 25): 'blue', (17, 25): 'blue', (1, 4): 'blue', (1, 5): 'blue', (1, 6):
'blue'}), (12, {(32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black',
(32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black',
(26, 8): 'black', (25, 8): 'black', (24, 8): 'black', (38, 15): 'green', (33, 21): 'green', (31, 13):
'green', (20, 25): 'blue', (19, 25): 'blue', (18, 25): 'blue', (1, 3): 'blue', (1, 4): 'blue', (1, 5):
'blue'}), (15, {(31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10):
'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8):
'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21):
'green', (20, 25): 'blue', (19, 25): 'blue', (18, 25): 'blue', (1, 3): 'blue', (1, 4): 'blue', (1, 5):
'blue'}), (15, {(30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11):
'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8):
'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (31, 5): 'green', (38, 15):
'green', (33, 21): 'green', (21, 25): 'blue', (20, 25): 'blue', (19, 25): 'blue', (1, 2): 'blue', (1, 3):
'blue', (1, 4): 'blue'}), (15, {(30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black',
(32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8):
'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26, 8): 'black', (25, 8):
'black', (31, 5): 'green', (38, 15): 'green', (33, 21): 'green', (21, 25): 'blue', (20, 25): 'blue', (19,
25): 'blue', (1, 2): 'blue', (1, 3): 'blue', (1, 4): 'blue'}), (15, {(31, 14): 'black', (30, 14): 'black',
(30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10):
'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8):
'black', (27, 8): 'black', (26, 8): 'black', (25, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21):
'green', (22, 25): 'blue', (21, 25): 'blue', (20, 25): 'blue', (1, 1): 'blue', (1, 2): 'blue', (1, 3):
'blue'}), (15, {(32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13):
'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32,
8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (26,
8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21): 'green', (22, 25): 'blue', (21, 25): 'blue', (20,
25): 'blue', (1, 1): 'blue', (1, 2): 'blue', (1, 3): 'blue'}), (15, {(33, 14): 'black', (32, 14): 'black',
(31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12):
'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8):
'black', (29, 8): 'black', (28, 8): 'black', (27, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21):
'green', (23, 25): 'blue', (22, 25): 'blue', (21, 25): 'blue', (1, 0): 'blue', (1, 1): 'blue', (1, 2):
'blue'}), (15, {(34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14):
```

```
'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32,
10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (28,
8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21): 'green', (23, 25): 'blue', (22, 25): 'blue', (21,
25): 'blue', (1, 0): 'blue', (1, 1): 'blue', (1, 2): 'blue'}), (15, {(35, 14): 'black', (34, 14): 'black',
(33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13):
'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (32,
8): 'black', (31, 8): 'black', (30, 8): 'black', (29, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33,
21): 'green', (24, 25): 'blue', (23, 25): 'blue', (22, 25): 'blue', (1, 0): 'blue', (1, 1): 'blue'}), (15,
{(36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14):
'black', (30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32,
11): 'black', (32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 8): 'black', (30, 8): 'black', (31,
5): 'green', (38, 15): 'green', (33, 21): 'green', (24, 25): 'blue', (23, 25): 'blue', (22, 25): 'blue', (1,
0): 'blue', (1, 1): 'blue'}), (15, {(37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14):
'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31,
13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black',
(32, 8): 'black', (31, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21): 'green', (25, 25): 'blue',
(24, 25): 'blue', (23, 25): 'blue', (1, 0): 'blue'}), (15, {(38, 14): 'black', (37, 14): 'black', (36, 14):
'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30,
14): 'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black',
(32, 10): 'black', (32, 9): 'black', (32, 8): 'black', (31, 5): 'green', (38, 15): 'green', (33, 21):
'green', (25, 25): 'blue', (24, 25): 'blue', (23, 25): 'blue', (1, 0): 'blue'}), (19, {(38, 15): 'black',
(38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14):
'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (32,
13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (31, 5): 'green',
(33, 21): 'green', (4, 19): 'green', (26, 25): 'blue', (25, 25): 'blue', (24, 25): 'blue'}), (19, {(38, 16):
'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34,
14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black',
(31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9):
'black', (31, 5): 'green', (33, 21): 'green', (4, 19): 'green', (26, 25): 'blue', (25, 25): 'blue', (24, 25):
'blue', (11, 29): 'blue', (11, 28): 'blue', (11, 27): 'blue'}), (19, {(38, 17): 'black', (38, 16): 'black',

(38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14):
'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31,
13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black',
(31, 5): 'green', (33, 21): 'green', (4, 19): 'green', (27, 25): 'blue', (26, 25): 'blue', (25, 25): 'blue',
(11, 29): 'blue', (11, 28): 'blue'}), (19, {(38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38,
15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black',
(33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13):
'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (32, 9): 'black', (31,
5): 'green', (33, 21): 'green', (4, 19): 'green', (27, 25): 'blue', (26, 25): 'blue', (25, 25): 'blue', (11,
29): 'blue', (11, 28): 'blue'}), (19, {(38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16):
'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34,
14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black',
(31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (32, 10): 'black', (31, 5):
'green', (33, 21): 'green', (4, 19): 'green', (28, 25): 'blue', (27, 25): 'blue', (26, 25): 'blue', (11, 29):
'blue'}), (19, {(38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16):
'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34,
14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black',
(31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (32, 11): 'black', (31, 5): 'green', (33, 21):
'green', (4, 19): 'green', (28, 25): 'blue', (27, 25): 'blue', (26, 25): 'blue', (11, 29): 'blue'}), (19,
{(38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16):
'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34,
14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black',
(31, 13): 'black', (32, 13): 'black', (32, 12): 'black', (31, 5): 'green', (33, 21): 'green', (4, 19):
'green', (29, 25): 'blue', (28, 25): 'blue', (27, 25): 'blue', (5, 25): 'blue', (4, 25): 'blue', (3, 25):
'blue'}), (19, {(38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18):
'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36,
14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black',
(30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (32, 13): 'black', (31, 5): 'green', (33, 21):
'green', (4, 19): 'green', (29, 25): 'blue', (28, 25): 'blue', (27, 25): 'blue', (5, 25): 'blue', (4, 25):
'blue', (3, 25): 'blue'}), (19, {(37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black',
(38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14):
'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32,
14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 13): 'black', (31, 5): 'green',
(33, 21): 'green', (4, 19): 'green', (30, 25): 'blue', (29, 25): 'blue', (28, 25): 'blue', (6, 25): 'blue',
(5, 25): 'blue', (4, 25): 'blue'}), (19, {(36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21):
'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38,
15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black',
```

```
(33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black', (30, 13): 'black', (31, 5):
'green', (33, 21): 'green', (4, 19): 'green', (30, 25): 'blue', (29, 25): 'blue', (28, 25): 'blue', (6, 25):
'blue', (5, 25): 'blue', (4, 25): 'blue'}), (19, {(35, 22): 'black', (36, 22): 'black', (37, 22): 'black',
(38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17):
'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35,
14): 'black', (34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14): 'black', (30, 14): 'black',
(31, 5): 'green', (33, 21): 'green', (4, 19): 'green', (31, 25): 'blue', (30, 25): 'blue', (29, 25): 'blue',
(7, 25): 'blue', (6, 25): 'blue', (5, 25): 'blue'}), (19, {(34, 22): 'black', (35, 22): 'black', (36, 22):
'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38,
18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black',
(36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32, 14): 'black', (31, 14):
'black', (31, 5): 'green', (33, 21): 'green', (4, 19): 'green', (31, 25): 'blue', (30, 25): 'blue', (29, 25):
'blue', (7, 25): 'blue', (6, 25): 'blue', (5, 25): 'blue'}), (19, {(33, 22): 'black', (34, 22): 'black', (35,
22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black',
(38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14):
'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (32,
14): 'black', (31, 5): 'green', (33, 21): 'green', (4, 19): 'green', (32, 25): 'blue', (31, 25): 'blue', (30,
25): 'blue', (8, 25): 'blue', (7, 25): 'blue', (6, 25): 'blue'}), (23, {(33, 21): 'black', (33, 22): 'black',
(34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21):
'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38,
15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black',
(33, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19): 'green', (32, 25): 'blue', (31, 25): 'blue',
(30, 25): 'blue', (8, 25): 'blue', (7, 25): 'blue', (6, 25): 'blue'}), (23, {(32, 21): 'black', (33, 21):
'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38,
22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black',
(38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14):
'black', (34, 14): 'black', (33, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19): 'green', (33,
25): 'blue', (32, 25): 'blue', (31, 25): 'blue', (9, 25): 'blue', (8, 25): 'blue', (7, 25): 'blue'}), (23,
{(31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22):
'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38,
19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black',

(37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (31, 5):
'green', (33, 20): 'green', (4, 19): 'green', (33, 25): 'blue', (32, 25): 'blue', (31, 25): 'blue', (9, 25):
'blue', (8, 25): 'blue', (7, 25): 'blue'}), (23, {(30, 21): 'black', (31, 21): 'black', (32, 21): 'black',
(33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22):
'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38,
17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black',
(35, 14): 'black', (34, 14): 'black', (33, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19):
'green', (34, 25): 'blue', (33, 25): 'blue', (32, 25): 'blue', (10, 25): 'blue', (9, 25): 'blue', (8, 25):
'blue'}), (23, {(29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21):
'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38,
22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black',
(38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14):
'black', (34, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19): 'green', (34, 25): 'blue', (33,
25): 'blue', (32, 25): 'blue', (10, 25): 'blue', (9, 25): 'blue', (8, 25): 'blue'}), (23, {(28, 21): 'black',
(29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22):
'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38,
21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black',
(38, 15): 'black', (38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (35, 14): 'black', (31, 5):
'green', (33, 20): 'green', (4, 19): 'green', (35, 25): 'blue', (34, 25): 'blue', (33, 25): 'blue', (11, 25):
'blue', (10, 25): 'blue', (9, 25): 'blue'}), (23, {(27, 21): 'black', (28, 21): 'black', (29, 21): 'black',
(30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22):
'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38,
20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black',
(38, 14): 'black', (37, 14): 'black', (36, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19):
'green', (35, 25): 'blue', (34, 25): 'blue', (33, 25): 'blue', (11, 25): 'blue', (10, 25): 'blue', (9, 25):
'blue'}), (23, {(27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21):
'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35,
22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black',
(38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14):
'black', (37, 14): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19): 'green', (36, 25): 'blue', (35,
25): 'blue', (34, 25): 'blue', (12, 25): 'blue', (11, 25): 'blue', (10, 25): 'blue'}), (23, {(28, 20):
'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31,
21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black',
(36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19):
'black', (38, 18): 'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (38, 14): 'black', (31,
5): 'green', (33, 20): 'green', (4, 19): 'green', (36, 25): 'blue', (35, 25): 'blue', (34, 25): 'blue', (12,
```

```
25): 'blue', (11, 25): 'blue', (10, 25): 'blue'}), (23, {(29, 20): 'black', (28, 20): 'black', (27, 20):
'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32,
21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black',
(37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18):
'black', (38, 17): 'black', (38, 16): 'black', (38, 15): 'black', (31, 5): 'green', (33, 20): 'green', (4,
19): 'green', (37, 25): 'blue', (36, 25): 'blue', (35, 25): 'blue', (13, 25): 'blue', (12, 25): 'blue', (11,
25): 'blue'}), (23, {(30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21):
'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33,
21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black',
(38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17):
'black', (38, 16): 'black', (31, 5): 'green', (33, 20): 'green', (4, 19): 'green', (37, 25): 'blue', (36,
25): 'blue', (35, 25): 'blue', (13, 25): 'blue', (12, 25): 'blue', (11, 25): 'blue'}), (23, {(31, 20):
'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28,
21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black',
(33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22):
'black', (38, 21): 'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (38, 17): 'black', (31,
5): 'green', (33, 20): 'green', (4, 19): 'green', (38, 25): 'blue', (37, 25): 'blue', (36, 25): 'blue', (14,
25): 'blue', (13, 25): 'blue', (12, 25): 'blue'}), (23, {(32, 20): 'black', (31, 20): 'black', (30, 20):
'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29,
21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black',
(34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21):
'black', (38, 20): 'black', (38, 19): 'black', (38, 18): 'black', (31, 5): 'green', (33, 20): 'green', (4,
19): 'green', (38, 25): 'blue', (37, 25): 'blue', (36, 25): 'blue', (14, 25): 'blue', (13, 25): 'blue', (12,
25): 'blue'}), (27, {(33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20):
'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30,
21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black',
(35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20):
'black', (38, 19): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (38,
25): 'blue', (37, 25): 'blue', (15, 25): 'blue', (14, 25): 'blue', (13, 25): 'blue'}), (27, {(34, 20):
'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28,
20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black',
(31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22):

'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38,
19): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (38, 25): 'blue', (37,
25): 'blue', (15, 25): 'blue', (14, 25): 'blue', (13, 25): 'blue'}), (27, {(35, 20): 'black', (34, 20):
'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28,
20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black',
(31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22):
'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38,
19): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (38, 25): 'blue', (16,
25): 'blue', (15, 25): 'blue', (14, 25): 'blue'}), (27, {(36, 20): 'black', (35, 20): 'black', (34, 20):
'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28,
20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black',
(31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22):
'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38, 20): 'black', (38,
19): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (38, 25): 'blue', (16,
25): 'blue', (15, 25): 'blue', (14, 25): 'blue'}), (27, {(37, 20): 'black', (36, 20): 'black', (35, 20):
'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29,
20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black',
(30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22):
'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (38,
20): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (17, 25): 'blue', (16,
25): 'blue', (15, 25): 'blue'}), (27, {(38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20):
'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29,
20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black',
(30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22):
'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (38, 21): 'black', (31,
5): 'green', (29, 18): 'green', (4, 19): 'green', (39, 25): 'blue', (17, 25): 'blue', (16, 25): 'blue', (15,
25): 'blue'}), (27, {(38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20):
'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29,
20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black',
(30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22):
'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (38, 22): 'black', (31, 5): 'green', (29,
18): 'green', (4, 19): 'green', (18, 25): 'blue', (17, 25): 'blue', (16, 25): 'blue', (5, 28): 'blue', (4,
28): 'blue', (3, 28): 'blue'}), (27, {(38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20):
'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31,
20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black',
```

```
(28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21):
'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36, 22): 'black', (37, 22): 'black', (31,
5): 'green', (29, 18): 'green', (4, 19): 'green', (18, 25): 'blue', (17, 25): 'blue', (16, 25): 'blue', (5,
28): 'blue', (4, 28): 'blue', (3, 28): 'blue'}), (27, {(38, 17): 'black', (38, 18): 'black', (38, 19):
'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33,
20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black',
(27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21):
'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22): 'black', (35, 22): 'black', (36,
22): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (19, 25): 'blue', (18, 25): 'blue', (17,
25): 'blue', (6, 28): 'blue', (5, 28): 'blue', (4, 28): 'blue'}), (27, {(38, 16): 'black', (38, 17): 'black',
(38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20):
'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29,
20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black',
(30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (34, 22):
'black', (35, 22): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (19, 25): 'blue', (18,
25): 'blue', (17, 25): 'blue', (6, 28): 'blue', (5, 28): 'blue', (4, 28): 'blue'}), (27, {(38, 15): 'black',
(38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20):
'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31,
20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black',
(28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21):
'black', (33, 22): 'black', (34, 22): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (20,
25): 'blue', (19, 25): 'blue', (18, 25): 'blue', (7, 28): 'blue', (6, 28): 'blue', (5, 28): 'blue'}), (27,
{(38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19):
'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33,
20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black',
(27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21):
'black', (32, 21): 'black', (33, 21): 'black', (33, 22): 'black', (31, 5): 'green', (29, 18): 'green', (4,
19): 'green', (20, 25): 'blue', (19, 25): 'blue', (18, 25): 'blue', (7, 28): 'blue', (6, 28): 'blue', (5,
28): 'blue'}), (27, {(37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17):
'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35,
20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black',
(29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21):
'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (33, 21): 'black', (31, 5): 'green', (29,

18): 'green', (4, 19): 'green', (21, 25): 'blue', (20, 25): 'blue', (19, 25): 'blue', (8, 28): 'blue', (7,
28): 'blue', (6, 28): 'blue'}), (27, {(36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15):
'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37,
20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black',
(31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21):
'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31, 21): 'black', (32, 21): 'black', (31,
5): 'green', (29, 18): 'green', (4, 19): 'green', (21, 25): 'blue', (20, 25): 'blue', (19, 25): 'blue', (8,
28): 'blue', (7, 28): 'blue', (6, 28): 'blue'}), (27, {(35, 14): 'black', (36, 14): 'black', (37, 14):
'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38,
19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black',
(33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20):
'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21): 'black', (30, 21): 'black', (31,
21): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (22, 25): 'blue', (21, 25): 'blue', (20,
25): 'blue', (9, 28): 'blue', (8, 28): 'blue', (7, 28): 'blue'}), (27, {(34, 14): 'black', (35, 14): 'black',
(36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17):
'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35,
20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black',
(29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (29, 21):
'black', (30, 21): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (22, 25): 'blue', (21,
25): 'blue', (20, 25): 'blue', (9, 28): 'blue', (8, 28): 'blue', (7, 28): 'blue'}), (27, {(33, 14): 'black',
(34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15):
'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37,
20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black',
(31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21):
'black', (28, 21): 'black', (29, 21): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (23,
25): 'blue', (22, 25): 'blue', (21, 25): 'blue', (10, 28): 'blue', (9, 28): 'blue', (8, 28): 'blue'}), (27,
{(32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14):
'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38,
19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black',
(33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20):
'black', (27, 20): 'black', (27, 21): 'black', (28, 21): 'black', (31, 5): 'green', (29, 18): 'green', (4,
19): 'green', (23, 25): 'blue', (22, 25): 'blue', (21, 25): 'blue', (10, 28): 'blue', (9, 28): 'blue', (8,
28): 'blue'}), (27, {(31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14):
'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38,
```

```
17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black',
(35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20):
'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (27, 21): 'black', (31, 5): 'green', (29,
18): 'green', (4, 19): 'green', (24, 25): 'blue', (23, 25): 'blue', (22, 25): 'blue', (11, 28): 'blue', (10,
28): 'blue', (9, 28): 'blue'}), (27, {(30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14):
'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38,
15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black',
(37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20):
'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28, 20): 'black', (27, 20): 'black', (31,
5): 'green', (29, 18): 'green', (4, 19): 'green', (24, 25): 'blue', (23, 25): 'blue', (22, 25): 'blue', (11,
28): 'blue', (10, 28): 'blue', (9, 28): 'blue'}), (27, {(30, 15): 'black', (30, 14): 'black', (31, 14):
'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37,
14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black',
(38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20):
'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30, 20): 'black', (29, 20): 'black', (28,
20): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (25, 25): 'blue', (24, 25): 'blue', (23,
25): 'blue', (12, 28): 'blue', (11, 28): 'blue', (10, 28): 'blue'}), (27, {(30, 16): 'black', (30, 15):
'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35,
14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black',
(38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20):
'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (30,
20): 'black', (29, 20): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green', (25, 25): 'blue',
(24, 25): 'blue', (23, 25): 'blue', (12, 28): 'blue', (11, 28): 'blue', (10, 28): 'blue'}), (27, {(30, 17):
'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33,
14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black',
(38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20):
'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32,
20): 'black', (31, 20): 'black', (30, 20): 'black', (31, 5): 'green', (29, 18): 'green', (4, 19): 'green',
(26, 25): 'blue', (25, 25): 'blue', (24, 25): 'blue', (13, 28): 'blue', (12, 28): 'blue', (11, 28): 'blue'}),
(27, {(30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31,
14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black',
(37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18):
'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34,
20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 20): 'black', (31, 5): 'green', (29, 18): 'green',

(4, 19): 'green', (26, 25): 'blue', (25, 25): 'blue', (24, 25): 'blue', (13, 28): 'blue', (12, 28): 'blue',
(11, 28): 'blue'}), (32, {(29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30,
15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black',
(35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16):
'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36,
20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 5): 'green',
(4, 19): 'green', (20, 21): 'green', (27, 25): 'blue', (26, 25): 'blue', (25, 25): 'blue', (14, 28): 'blue',
(13, 28): 'blue', (12, 28): 'blue'}), (32, {(29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30,
17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black',
(33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14):
'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38,
20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33, 20): 'black',
(32, 20): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (27, 25): 'blue', (26, 25): 'blue',
(25, 25): 'blue', (14, 28): 'blue', (13, 28): 'blue', (12, 28): 'blue'}), (32, {(29, 20): 'black', (29, 19):
'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30,
14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black',
(36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17):
'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35,
20): 'black', (34, 20): 'black', (33, 20): 'black', (32, 20): 'black', (31, 5): 'green', (4, 19): 'green',
(20, 21): 'green', (28, 25): 'blue', (27, 25): 'blue', (26, 25): 'blue', (15, 28): 'blue', (14, 28): 'blue',
(13, 28): 'blue'}), (32, {(29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30,
18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black',
(32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14):
'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38,
19): 'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black',
(33, 20): 'black', (32, 20): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (28, 25):
'blue', (27, 25): 'blue', (26, 25): 'blue', (15, 28): 'blue', (14, 28): 'blue', (13, 28): 'blue'}), (32,
{(29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18):
'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32,
14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black',
(38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19):
'black', (38, 20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (33,
20): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (29, 25): 'blue', (28, 25): 'blue', (27,
```

```
25): 'blue', (16, 28): 'blue', (15, 28): 'blue', (14, 28): 'blue'}), (32, {(29, 23): 'black', (29, 22):
'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30,
17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black',
(33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14):
'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38,
20): 'black', (37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (34, 20): 'black', (31, 5): 'green',
(4, 19): 'green', (20, 21): 'green', (29, 25): 'blue', (28, 25): 'blue', (27, 25): 'blue', (16, 28): 'blue',
(15, 28): 'blue', (14, 28): 'blue'}), (32, {(28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29,
21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black',
(30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14):
'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38,
15): 'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black',
(37, 20): 'black', (36, 20): 'black', (35, 20): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21):
'green', (30, 25): 'blue', (29, 25): 'blue', (28, 25): 'blue', (17, 28): 'blue', (16, 28): 'blue', (15, 28):
'blue'}), (32, {(27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21):
'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30,
16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black',
(34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15):
'black', (38, 16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37,
20): 'black', (36, 20): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (30, 25): 'blue',
(29, 25): 'blue', (28, 25): 'blue', (17, 28): 'blue', (16, 28): 'blue', (15, 28): 'blue'}), (32, {(26, 23):
'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29,
20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black',
(30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14):
'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38,
16): 'black', (38, 17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (37, 20): 'black',
(31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (31, 25): 'blue', (30, 25): 'blue', (29, 25): 'blue',
(18, 28): 'blue', (17, 28): 'blue', (16, 28): 'blue'}), (32, {(25, 23): 'black', (26, 23): 'black', (27, 23):
'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29,
19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black',
(30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14):
'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38,
17): 'black', (38, 18): 'black', (38, 19): 'black', (38, 20): 'black', (31, 5): 'green', (4, 19): 'green',
(20, 21): 'green', (31, 25): 'blue', (30, 25): 'blue', (29, 25): 'blue', (18, 28): 'blue', (17, 28): 'blue',
(16, 28): 'blue'}), (32, {(24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28,

23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black',
(29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14):
'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36,
14): 'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black',
(38, 18): 'black', (38, 19): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (32, 25):
'blue', (31, 25): 'blue', (30, 25): 'blue', (19, 28): 'blue', (18, 28): 'blue', (17, 28): 'blue'}), (32,
{(23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23):
'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29,
18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black',
(31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14):
'black', (37, 14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (38,
18): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (32, 25): 'blue', (31, 25): 'blue', (30,
25): 'blue', (19, 28): 'blue', (18, 28): 'blue', (17, 28): 'blue'}), (32, {(22, 23): 'black', (23, 23):
'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29,
23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black',
(30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14):
'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37,
14): 'black', (38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (38, 17): 'black', (31, 5): 'green',
(4, 19): 'green', (20, 21): 'green', (33, 25): 'blue', (32, 25): 'blue', (31, 25): 'blue', (20, 28): 'blue',
(19, 28): 'blue', (18, 28): 'blue'}), (32, {(21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24,
23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black',
(29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18):
'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32,
14): 'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black',
(38, 14): 'black', (38, 15): 'black', (38, 16): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21):
'green', (33, 25): 'blue', (32, 25): 'blue', (31, 25): 'blue', (20, 28): 'blue', (19, 28): 'blue', (18, 28):
'blue'}), (32, {(21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23):
'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29,
22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black',
(30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14):
'black', (33, 14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38,
14): 'black', (38, 15): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (34, 25): 'blue',
```

```
(33, 25): 'blue', (32, 25): 'blue', (21, 28): 'blue', (20, 28): 'blue', (19, 28): 'blue'}), (32, {(21, 21):
'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25,
23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black',
(29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17):
'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33,
14): 'black', (34, 14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (38, 14): 'black',
(31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (34, 25): 'blue', (33, 25): 'blue', (32, 25): 'blue',
(21, 28): 'blue', (20, 28): 'blue', (19, 28): 'blue'}), (32, {(21, 20): 'black', (21, 21): 'black', (21, 22):
'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26,
23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black',
(29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16):
'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34,
14): 'black', (35, 14): 'black', (36, 14): 'black', (37, 14): 'black', (31, 5): 'green', (4, 19): 'green',
(20, 21): 'green', (35, 25): 'blue', (34, 25): 'blue', (33, 25): 'blue', (22, 28): 'blue', (21, 28): 'blue',
(20, 28): 'blue'}), (32, {(20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21,
23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black',
(27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20):
'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30,
15): 'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black',
(35, 14): 'black', (36, 14): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (35, 25):
'blue', (34, 25): 'blue', (33, 25): 'blue', (22, 28): 'blue', (21, 28): 'blue', (20, 28): 'blue'}), (32,
{(19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23):
'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27,
23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black',
(29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15):
'black', (30, 14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (35,
14): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (36, 25): 'blue', (35, 25): 'blue', (34,
25): 'blue', (23, 28): 'blue', (22, 28): 'blue', (21, 28): 'blue'}), (32, {(19, 21): 'black', (19, 20):
'black', (20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23): 'black', (22,
23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black',
(28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19):
'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30,
14): 'black', (31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (34, 14): 'black', (31, 5): 'green',
(4, 19): 'green', (20, 21): 'green', (36, 25): 'blue', (35, 25): 'blue', (34, 25): 'blue', (23, 28): 'blue',
(22, 28): 'blue', (21, 28): 'blue'}), (32, {(19, 22): 'black', (19, 21): 'black', (19, 20): 'black', (20,
20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black',

(23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23):
'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29,
18): 'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black',
(31, 14): 'black', (32, 14): 'black', (33, 14): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21):
'green', (37, 25): 'blue', (36, 25): 'blue', (35, 25): 'blue', (24, 28): 'blue', (23, 28): 'blue', (22, 28):
'blue'}), (32, {(19, 23): 'black', (19, 22): 'black', (19, 21): 'black', (19, 20): 'black', (20, 20):
'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23,
23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black',
(29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18):
'black', (30, 18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31,
14): 'black', (32, 14): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (37, 25): 'blue',
(36, 25): 'blue', (35, 25): 'blue', (24, 28): 'blue', (23, 28): 'blue', (22, 28): 'blue'}), (32, {(19, 24):
'black', (19, 23): 'black', (19, 22): 'black', (19, 21): 'black', (19, 20): 'black', (20, 20): 'black', (21,
20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black',
(24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23):
'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30,
18): 'black', (30, 17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 14): 'black',
(31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (38, 25): 'blue', (37, 25): 'blue', (36, 25): 'blue',
(25, 28): 'blue', (24, 28): 'blue', (23, 28): 'blue'}), (32, {(19, 25): 'black', (19, 24): 'black', (19, 23):
'black', (19, 22): 'black', (19, 21): 'black', (19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21,
21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black',
(25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22):
'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30,
17): 'black', (30, 16): 'black', (30, 15): 'black', (30, 14): 'black', (31, 5): 'green', (4, 19): 'green',
(20, 21): 'green', (38, 25): 'blue', (37, 25): 'blue', (36, 25): 'blue', (25, 28): 'blue', (24, 28): 'blue',
(23, 28): 'blue'}), (32, {(19, 26): 'black', (19, 25): 'black', (19, 24): 'black', (19, 23): 'black', (19,
22): 'black', (19, 21): 'black', (19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21, 21): 'black',
(21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23):
'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29,
21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black',
(30, 16): 'black', (30, 15): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (39, 25):
```

```
'blue', (38, 25): 'blue', (37, 25): 'blue', (26, 28): 'blue', (25, 28): 'blue', (24, 28): 'blue'}), (32,
{(19, 27): 'black', (19, 26): 'black', (19, 25): 'black', (19, 24): 'black', (19, 23): 'black', (19, 22):
'black', (19, 21): 'black', (19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21,
22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black',
(26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21):
'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (30,
16): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (39, 25): 'blue', (38, 25): 'blue', (37,
25): 'blue', (26, 28): 'blue', (25, 28): 'blue', (24, 28): 'blue'}), (32, {(19, 28): 'black', (19, 27):
'black', (19, 26): 'black', (19, 25): 'black', (19, 24): 'black', (19, 23): 'black', (19, 22): 'black', (19,
21): 'black', (19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black',
(21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23):
'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29,
20): 'black', (29, 19): 'black', (29, 18): 'black', (30, 18): 'black', (30, 17): 'black', (31, 5): 'green',
(4, 19): 'green', (20, 21): 'green', (39, 25): 'blue', (38, 25): 'blue', (27, 28): 'blue', (26, 28): 'blue',
(25, 28): 'blue'}), (32, {(19, 29): 'black', (19, 28): 'black', (19, 27): 'black', (19, 26): 'black', (19,
25): 'black', (19, 24): 'black', (19, 23): 'black', (19, 22): 'black', (19, 21): 'black', (19, 20): 'black',
(20, 20): 'black', (21, 20): 'black', (21, 21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23):
'black', (23, 23): 'black', (24, 23): 'black', (25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28,
23): 'black', (29, 23): 'black', (29, 22): 'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black',
(29, 18): 'black', (30, 18): 'black', (31, 5): 'green', (4, 19): 'green', (20, 21): 'green', (39, 25):
'blue', (38, 25): 'blue', (27, 28): 'blue', (26, 28): 'blue', (25, 28): 'blue'}), (32, {(19, 29): 'black',
(19, 28): 'black', (19, 27): 'black', (19, 26): 'black', (19, 25): 'black', (19, 24): 'black', (19, 23):
'black', (19, 22): 'black', (19, 21): 'black', (19, 20): 'black', (20, 20): 'black', (21, 20): 'black', (21,
21): 'black', (21, 22): 'black', (21, 23): 'black', (22, 23): 'black', (23, 23): 'black', (24, 23): 'black',
(25, 23): 'black', (26, 23): 'black', (27, 23): 'black', (28, 23): 'black', (29, 23): 'black', (29, 22):
'black', (29, 21): 'black', (29, 20): 'black', (29, 19): 'black', (29, 18): 'black', (31, 5): 'green', (4,
19): 'green', (20, 21): 'green', (39, 25): 'blue', (28, 28): 'blue', (27, 28): 'blue', (26, 28): 'blue'})]
```

7 snake.py

```python
##################################################################
# FILE : snake.py
# WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
# EXERCISE : intro2cs1 ex10 2023
# DESCRIPTION: Implements the Snake for the Snake Game
# STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
# WEB PAGES I USED: N/A
# NOTES: N/A
##################################################################

from typing import List
from common import BaseDynamicGameObject, Coordinate, Direction

class Snake(BaseDynamicGameObject):
    """
    Represents the Snake Game Object in the game.
    The object is intended to be placed on a board, although it's not mandatory.
    """

    def __init__(self, location: Coordinate, length: int = 3) -> None:
        """
        Creates a new Snake with its head in the given coordinate and initial given length.
        The Snake starts at an upwards orientation, with the starting coordinates being
        from head to the tail perpendicular to the X-Axis.
        """
        # The coordinates which the snake occupies, the order of this list is
        #   integral to the operation of the object
        super().__init__(Direction.UP, self._get_initial_position(location, length), "black")
        self._expansion = 0
        self._to_split = None

    def move(self) -> bool:
        """
        Moving the Snake in the set direction.
        To set the direction use change_direction.

        If the Snake is during expansion phase, it will not empty
        the coordinates it occupies until the expansion phase is done.
        (e.g. the tail is not removed during this phase)
        """
        if self._to_split is not None:
            self._coordinates = self._coordinates[:self._to_split]
            self._to_split = None

        new_coordinate = self.movement_requirements()
        self._coordinates.insert(0, new_coordinate) # Adding the new head

        # If expanding the snake length is not necessary, remove the tail element
        if 0 == self._expansion:
            self._coordinates.pop() # Removing the last element (the tail)
        else: # Otherwise, expand by 1 and decrement the expanion ratio
            self._expansion -= 1

        # Moving the snake cannot fail
        return True

    def change_direction(self, direction: Direction) -> bool:
        """
        Changes the direction the snake is moving to.
```

```
60          If the direction is invalid False is returned, True otherwise.
61          """
62          # Checking if the direction is in the valid directions
63          #   for the current snake orientation
64          if direction not in self._get_valid_directions():
65              return False
66
67          # Updating the current direction
68          self._current_direction = direction
69          return True
70
71      def expand(self, amount: int) -> None:
72          """
73          Ordering the Snake to expand on the next moves by the given amount.
74          Expansion process is taking 'amount' of calls to move to fully complete.
75          """
76          self._expansion += amount
77
78      def split(self, coordinate: Coordinate) -> bool:
79          """
80          Ordering the Snake to split part of its body from the given
81          coordinate (included) to the tail
82          Returns True whether the split would be successful, False otherwise
83          """
84          self._to_split = self._coordinates.index(coordinate)
85          return not (self._to_split in [0,1])
86
87      def _get_valid_directions(self) -> List[Direction]:
88          """
89          Returns the valid directions for the snake to move at its current orientation.
90          It is not permitted for the snake to move down (up) if it's oriented up (down)
91          or to move left (right) if it's oriented right (left).
92          """
93          if (Direction.LEFT == self._current_direction) or \
94                  (Direction.RIGHT == self._current_direction):
95              return [Direction.UP, Direction.DOWN, self._current_direction]
96          else: # Up or Down
97              return [Direction.RIGHT, Direction.LEFT, self._current_direction]
98
99      def _get_initial_position(self, head_location: Coordinate, length: int) -> List[Coordinate]:
100         """
101         Returns the initial coordinates considering the location of the head and length.
102         The function expects that the snake would be oriented UPWARDS.
103         """
104         # Initializing the starting coordinates of the snake to be under the head location
105         return [Coordinate(head_location.row-index, head_location.column)
106                     for index in range(length)]
```

8 snake game.py

```
1  ######################################################################
2  # FILE : snake_game.py
3  # WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
4  # EXERCISE : intro2cs1 ex10 2023
5  # DESCRIPTION: Implements the primary game functionality
6  # STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
7  # WEB PAGES I USED: N/A
8  # NOTES: N/A
9  ######################################################################
10
11 # External imports
12 import math
13 from typing import Optional
14 from game_display import GameDisplay
15 from game_utils import get_random_apple_data, get_random_wall_data
16
17 # Internal imports
18 from common import Coordinate, BaseGameObject, Direction
19 from snake import Snake
20 from board import Board
21 from apple import Apple
22 from wall import Wall
23
24 class SnakeGame:
25     """
26     The primary game object for Snake
27     """
28
29     def __init__(
30             self,
31             board: Board,
32             snake: Optional[Snake],
33             max_apples: int,
34             max_walls: int,
35             max_rounds: int) -> None:
36         """
37         Creates a new Snake Game with the given Board and Snake.
38         max_apples and max_walls specify the maximum amount of
39             apples and walls to be on the wall at a given time.
40         max_rounds specifies when the game will terminate automatically
41             if it didn't terminate naturally.
42         """
43
44         self._new_direction = None
45         self._snake = snake
46         self._board = board
47
48         # Apple info
49         self._max_apples = max_apples
50         self._current_apples = 0
51
52         # Wall info
53         self._max_walls = max_walls
54         self._current_walls = 0
55         self._score = 0
56         self._is_over = False
57         self._current_round = 0
58         self._max_rounds = max_rounds
59
```

```
60          # If we received a snake object add it to the board
61          if self._snake is not None:
62              self._board.add_game_object(self._snake)
63
64          # Edge case: Adding apples and walls on the Snake itself will cause
65          #   them not to generate in the first round
66          self._add_apples_and_walls()
67
68      def set_snake_direction(self, direction: Optional[Direction])-> None:
69          """
70          Setting the moving direction of the snake.
71          """
72          self._new_direction = direction
73
74      def add_points(self) -> None:
75          """
76          Adding to the score tally.
77          The points are calculated by the floor of the
78              square root of the snake's current length.
79          """
80          self._score += math.floor(math.sqrt(len(self._snake)))
81
82      def update_objects(self) -> None:
83          """
84          """
85          # Changing the direction of the snake if we have a snake
86          if self._snake is not None:
87              self._snake.change_direction(self._new_direction)
88
89          self._board.move_game_objects(self._interaction_callback,
90                                        self._out_of_bounds_callback)
91
92          self._add_apples_and_walls()
93
94      def _out_of_bounds_callback(self, source: BaseGameObject, off_board: bool) -> None:
95          """
96          Deals with all interactions of game objects and the game board.
97          Called from within Board.
98          """
99          if type(source) is Wall and off_board:
100             # Wall is completely off the board, we should remove it
101             self._board.remove_game_object(source)
102             self._current_walls -= 1
103
104         if type(source) is Snake:
105             # The snake met with the board boundries, we end the game
106             self._set_is_over()
107
108     def _interaction_callback(self, source: BaseGameObject, dest: BaseGameObject) -> None:
109         """
110         Deals with all interactions of the game objects with one another.
111         Called from within Board.
112         """
113         if type(dest) is Apple and type(source) is Snake:
114             # The snake with an apple, remove it,
115             #   increment the score and expand the snake
116             self.add_points()
117             self._board.remove_game_object(dest)
118             self._current_apples -= 1
119             self._snake.expand(3)
120
121         if type(dest) is Apple and type(source) is Wall:
122             # The wall met with an apple, destroy it
123             self._board.remove_game_object(dest)
124             self._current_apples -= 1
125
126         if type(dest) is Snake and type(source) is Wall:
127             # The wall met with a snake, split the snake
```

```
128             # If the split left the snake with 0 or 1 cells,
129             #   the game ends
130             if not dest.split(source.get_coordinates()[0]):
131                 self._set_is_over()
132 
133         # Ending the game if the snake hits a wall, or hit itself
134         if (type(dest) is Snake or type(dest) is Wall) and type(source) is Snake:
135             self._set_is_over()
136 
137     def _wall_move_callback(self) -> bool:
138         """
139         Deals with the conditions for the wall to move.
140         Currently the wall moves if and only if the round is even number.
141         """
142         return 0 == (self._current_round % 2)
143 
144     def _add_apples_and_walls(self) -> None:
145         """
146         Adds apples and walls to the wall, if any are missing.
147         The coordinates for each are generated randomly, if the
148             generated coordinates overlap with any other object on the game
149             board the addition fails and it is not tried again
150         """
151         # Adding walls if missing any
152         if self._max_walls > self._current_walls:
153             x_coord, y_coord, direction = get_random_wall_data()
154             if self._board.add_game_object(
155                 Wall(Coordinate.from_legacy_coordinate((x_coord, y_coord)),
156                      direction,
157                      self._wall_move_callback)):
158                 self._current_walls += 1
159 
160         # Adding apples if missing
161         if self._max_apples > self._current_apples:
162             if self._board.add_game_object(
163                 Apple(Coordinate.from_legacy_coordinate(get_random_apple_data()))):
164                 self._current_apples += 1
165 
166     def draw_board(self, gui: GameDisplay) -> None:
167         """
168         Drawing the current board on the screen and setting the score value
169         """
170         self._board.draw_board(gui)
171         gui.show_score(self._score)
172 
173     def end_round(self) -> None:
174         """
175         Finishing the current round
176         Placing apples and walls if needed, and incrementing the round counter
177         """
178         self._current_round += 1
179 
180     def _set_is_over(self) -> None:
181         """
182         Setting the is_over flag, signaling that the game should end
183         """
184         self._is_over = True
185 
186     def is_over(self) -> bool:
187         """
188         Returns if any of the end conditions were satisfied.
189         Possible ending routes:
190             1) The max round has been reached.
191             2) The Snake met with a wall or the board boundries
192             3) The Snake was split to an invalid length (currently 1 or 0)
193         """
194         return self._is_over or (self._current_round > self._max_rounds if self._max_rounds != -1 else False)
```

9 snake main.py

```python
######################################################################
# FILE : snake_main.py
# WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
# EXERCISE : intro2cs1 ex10 2023
# DESCRIPTION: Runs the Snake Game
# STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
# WEB PAGES I USED: N/A
# NOTES: N/A
######################################################################

# External imports
import argparse
from game_display import GameDisplay

# Internal imports
from snake_game import SnakeGame
from common import Coordinate
from board import Board
from snake import Snake

def _initialize_game(args: argparse.Namespace) -> SnakeGame:
    """
    Initializing the game with the given arguments.
    """
    board = Board(Coordinate(args.height, args.width))

    # If game is not on debug mode, create the snake
    snake = None
    if not args.debug:
        snake = Snake(Coordinate(args.height//2, args.width//2))

    return SnakeGame(board, snake, args.apples, args.walls, args.rounds)

def main_loop(gd: GameDisplay, args: argparse.Namespace) -> None:
    """
    Runs the Snake Game with the given arguments and Game Display.
    """
    game = _initialize_game(args)
    gd.show_score(0)

    # ROUND 0 STARTS HERE
    # No movements are made in the round 0
    game.draw_board(gd)
    game.end_round()
    gd.end_round()
    # ROUND 0 ENDS HERE

    # Begin primary game loop
    while not game.is_over():

        # Changing the direction of the snake if necessary
        game.set_snake_direction(gd.get_key_clicked())
        # Updating the objects on the game board
        game.update_objects()

        # Round finalization
        # Drawing the board and ending the round
        game.draw_board(gd)
        game.end_round()
```

```
60          gd.end_round()
61
62  if __name__ == "__main__":
63      print("You should run:\n"
64            "> python game_display.py")
```

10 wall.py

```
1  ##################################################################
2  # FILE : wall.py
3  # WRITERS : Nimrod Mallis, mallis ; Dor Kaiser, dor_kaiser
4  # EXERCISE : intro2cs1 ex10 2023
5  # DESCRIPTION: Implements the Wall for the Snake Game
6  # STUDENTS I DISCUSSED THE EXERCISE WITH: N/A
7  # WEB PAGES I USED: N/A
8  # NOTES: N/A
9  ##################################################################
10
11 from typing import List, Callable
12 from common import BaseDynamicGameObject, Coordinate, Direction
13
14 # Callback passed to the wall, it is called when the wall is requested to move
15 # and the callback should determine whether the movement should occur, so
16 # a True should be returned if the move is permitted, False otherwise
17 MoveCallback = Callable[[], bool]
18
19 class Wall(BaseDynamicGameObject):
20     """
21     Represents a wall in the Snake Game.
22     A wall is a moving object, moving perpendicular to the Y or X Axis,
23     depending on its starting orientation.
24     """
25
26     def __init__(
27         self, location: Coordinate, direction: Direction, move_callback: MoveCallback) -> None:
28         """
29         Creates a new wall at the given location and orientation.
30         The received location is a coordinate of where the middle of the wall lies.
31         The move_callback is called when the wall is requested to be moved.
32         """
33         super().__init__(direction, self._get_initial_position(location, direction), "blue")
34         self._move_callback = move_callback
35
36     def move(self) -> bool:
37         """
38         Moves the wall one step
39         The function calls the move callback received in the Ctor, and expects
40         that the callback return whether the move is permitted or not.
41         """
42         # Check if moving is permitted before actually moving
43         if self._move_callback():
44             self._coordinates.insert(0, self.movement_requirements()) # Adding the new head
45             self._coordinates.pop() # Removing the last element (the tail)
46             return True
47         return False # If no movement made, return so
48
49     def _get_initial_position(
50         self, mid_location: Coordinate, direction: Direction) -> List[Coordinate]:
51         """
52         Returns the initial position of the wall on the play board.
53         The wall occupies total of 3 cells in a board.
54         """
55         if direction == Direction.UP:
56             return [Coordinate(mid_location.row+row_index, mid_location.column)
57                         for row_index in range(1, -2, -1)]
58         elif direction == Direction.DOWN:
59             return [Coordinate(mid_location.row+row_index, mid_location.column)
```

```
60                      for row_index in range(-1, 2)]
61          elif direction == Direction.LEFT:
62              return [Coordinate(mid_location.row, mid_location.column+column_index)
63                      for column_index in range(-1, 2)]
64          elif direction == Direction.RIGHT:
65              return [Coordinate(mid_location.row, mid_location.column+column_index)
66                      for column_index in range(1, -2, -1)]
```