**Design Document**

-- 120090828 Lyu Minen SSE

Part I (Design):

1. Overview:

The game is composed of 3 objects: a snake, a monster and food items represented by a set of numbers from 1 to 9. In the figure shown above, the snake is represented by a sequence of squares where its head and its body are displayed in red and black colors respectively, while the monster by a purple square. The numbers are food items to be consumed by the snake. The goal of the game is to maneuver the snake within the game area in four directions (up, down, left and right), trying to consume all the food items while avoiding head-on collision with the monster. As each food item is consumed, the snake grows with its body lengthened in size equal to the value of the number being passed. While directing the movement of the snake you should avoid contact with the monster. Furthermore, the monster is also programmed to be motioned in the direction towards the head of the snake at a variable speed.

1. Date Model:

Snakes: turtles inside a list

Foods: turtles inside a list

Monster: turtle

1. Program Structure

Move Control: a function to control the movement of the snake

A function to control the movement of the monster (contains a function to find the direction)

Length of the snake: a function of draw the snakes

A function to extend the length

Judgement: define a function to check whether it eat the food and remove the food

A function to check whether monster touch the snake

A function to check whether the snakes extend fully

Food control: define a function to draw food

Contact count: define a function to check the number of contacts

Use the turtle to write the number of contacts

Time counting: a global variable to check the time

1. Processing Logic:

The whole process:

First draw the structure and the monster, foods, snakes and mark the time. Second, when the player clicks the screen, game starts. Third, monster and snakes and the column start to work. While moving, check whether they win or lose. The status, number of contacts and time can be shown. When game, print whether they win or lose.

1. motion the snake and monster:

I use a “for” loop to control the snakes. All parts of the body move to the position of the previous one. The movement starts from the last one. The directions of the head depend on the keyboard control.

For the monster, I create a function to find the direction. If the difference between the monster and the snake’s head is smallest, choose this direction. Its speed should be smaller than the snakes’.

1. extend the snake:

I use “colon” to copy the turtle in the snake’s body. And insert the new body segment into the previous list in the second position.

1. detect the body contact:

I use “for” loop to check the contact. If I find a contact, the loop break and wait for another time.

Part II function specification

1. draw snake:

This function is used to draw a snake. The length of the snake is a parameter. A list is used to store all turtles that stand for the snake’s body segment.

1. move snake:

This function is used to move the snake. The snake’s body moves from the last body segment in the special square if the game is “TRUE”. Each part moves to the previous part. The head’s direction depends on the keyboard.

1. right snake:

The snake’s head will change to east if it is not towards west. The status will change as well.

1. left snake:

The snake’s head will change to west if it is not towards east. The status will change as well.

1. up snake:

The snake’s head will change to north if it is not towards south. The status will change as well.

1. down snake:

The snake’s head will change to south if it is not towards north. The status will change as well.

1. draw mons:

This function creates a turtle called turtle.

1. on\_dir\_mons:

This function uses the difference between the angle from snake’s head to the monster and the fore direction to determine which direction to go. Absolute value is used here.

1. ini \_pos:

This function is used to find the position that haven’t been stored. The position is selected by random function. If it is not in the list g \_pos, it become a new position.

1. draw foods:

This function creates a list to store nine turtles. The turtles write number 1 to 9 to represent all the foods. The foods are placed randomly.

1. change snake:

The function is used to change the length of snakes. If the length change, the new turtle would be copied from the first body segment of the snakes. The new segments are inserted in the second place of the snakes list.

1. check contact:

The function is used to check whether the snakes body crash the monster. The function would check every segment in the snakes. If find one segment, the “for” loop would stops.

1. check \_eat:

The function is used to check whether the snake eats the food. If the distance between the snake’s head and the food is less than 15, the snakes would become longer. The food would be release as well.

1. Score:
2. game over:

This function is used to check whether the monster crash the snake’s head. If the distance is less than 15, the game is False status. The game would end. Monster would write “game over”.

1. game win:

This function would check whether the length of the snake reaches maximum and expend. If it does, game finishes. The monster would write “win”.

1. on \_Snake:

This function is used to control every movement of the snake. It first moves the snakes. Then check whether the snakes crush the monster or it fully extended.

1. on \_Mons:

The function is used to control every movement of the monster. If the game is “True”, first choose the proper direction then move forward.

1. on \_Write:

The function is used to update the contact, time and statues that is in the upper square. Three turtle would write the things depend on the current game status.

1. game \_start:

This is an entrance of the game. Once user click the screen, the time would be marked. Four direction functions would be connected to the keyboard control as well.

Part III output

