CS 111 – Introduction to Computer Science – Spring 2018

Programming Assignment #3

Caterpillar

Due Date: at 11:59pm on Saturday, April 21.

For this assignment, you will design and implement a program that requires the use of functions.

Remember that assignments are to be done individually, you should not share code, share design ideas, or look at anyone's code. If you need help, you may use the tutor, TA, or professor. If you have questions about this policy, refer to the syllabus or ask the professor.

Before getting started with the project, create a folder named `proj3` within your `U:\cs111` folder and use it to save your source file for this project.

Background

Beetle (in England) and Cootie (in America) are children's game with the goal of building a complete bug before your opponent. During each turn, a die is rolled to determine which part of the bug to add, with some restrictions. For instance, you can't place a head on a bug without a body. For this project you will build a version of the game called Caterpillar. The caterpillar has 2 body segments, 1 head, 2 antenna, and 6 legs, but you can't have any other body parts without a body segment, any antenna without a head, or more than 3 legs per body segment.

The classic build-a-beetle game
BEETLE
Ages 4 and up
2 to 4 Players
Mr Beetle's all in pieces. He waits for you to play his game.
Take a leaf and spin the spinner-he's keen to know the winner's name!
WARNING:
CHOKING HAZARD-Small parts.
Not for children under 3 years.
Cootie Games
Cootie

Beetle and Cootie

Program Description

You will implement a program `caterpillar.py` that allows for one player to "roll" a die in an attempt to build a caterpillar in at most 30 rolls. A roll of 1 corresponds to an antenna, 2 to a head, 3 to a body segment, 4 or 5 to a leg, and a 6 causes the loss of a turn. When the caterpillar is completed it will have two antenna, one head, two body segments, and six legs. However, you cannot have a head without a body or an antenna without a head, and you can only have 3 legs per body segment.

Coding this game all at once is doable, but complex, and would be difficult to check for errors. The game has been broken down into pieces, each which will be coded as one function. These include printing the parts of the bug, rolling the die, determining if the caterpillar is complete, determining if a new antenna, head, body segment, or leg can be added, and, finally, the running of the game.

Your game should have the following functions, and no main function or floating code. Pay attention to the order of the parameters. It would be best to code in the order of functions given for easiest testing. Use `testCaterpillar.py` to test each of your functions as you code them (comment out the test functions that you are not ready for in the main function).

- `drawCaterpillar(antenna, head, body, leg)` – Receives the number of body parts for the caterpillar and prints them for the user to see. Does not return anything.

  The completed caterpillar:

  ```

           \ /
            0
  * * *-* * *
  | | | | | |

  ```

  If there is one antenna, print the left one, and if there are two, print both of them. Print the head if there is one. If there is only one body segment, print the right one (the one closest to the head) without the dash, and if there are two, print both of them with the dash. Legs should be attached to the body segments, so if there is one, print the rightmost one, if there are two, print the rightmost two, etc. You can assume that the number of body parts follows the requirements of the game, so you do not need to check for any restrictions. Each body part should be printed in its final location (so the head should always be on the right, for instance). Print a blank line leading and following the pieces of the caterpillar.

  Note that printing a single backslash (\) requires a string with two backslashes (\\).

- `rollDie()` – Receives no parameters. Returns a random integer between 1 and 6.

- `isCompleteCaterpillar(antenna, head, body, leg)` – Receives the number of body parts for the caterpillar and returns `True` if the caterpillar is complete, and `False` otherwise.

- `useAntenna(antenna, head, body, leg)` – Receives the current number of body parts for the caterpillar and returns `True` if the antenna can be used, and `False` otherwise. To use the antenna, the caterpillar must already have a head and not have the maximum number of antenna. This function should also print the result to the player, by printing the one reason the antenna can't be used:

```
You can't have an antenna without a head.

You've already got antenna.
```

or announcing that the antenna will be added:

```
You got an antenna!
```

- `useHead(antenna, head, body, leg)` – Receives the current number of body parts for the caterpillar and returns `True` if the head can be used, and `False` otherwise. To use the head, the caterpillar must already have a body segment and not have a head. This function should also print the result to the player, by printing the one reason the head can't be used:

  ```
  You can't have a head without a body.

  You've already got a head.
  ```

  or announcing that the head will be added:

  ```
  You got a head!
  ```

- `useBody(antenna, head, body, leg)` – Receives the current number of body parts for the caterpillar and returns `True` if the body can be used, and `False` otherwise. To use the body, the caterpillar must not already have the maximum number of body segments. This function should also print the result to the player, by printing the one reason the body can't be used:

  ```
  You've already got your body segments.
  ```

  or announcing that the head will be added:

  ```
  You got a body!
  ```

- `useLeg(antenna, head, body, leg)` – Receives the current number of body parts for the caterpillar and returns `True` if the leg can be used, and `False` otherwise. To use the leg, the caterpillar must already have a body segment and less than 3 legs, or two body segments and less that the maximum number of legs. This function should also print the result to the player, by printing the one reason the leg can't be used:

  ```
  You need another body segment for this leg.
  ```

  or announcing that an antenna will be added:

  ```
  You got a leg!
  ```

- `playGame()` – Welcomes the player to the game and prints an example of a complete caterpillar.

```
********************************************
**** Welcome to the game of Caterpillar ****
********************************************

         \ /
          O
* * *-* * *
| | | | | |
```

While the player's caterpillar has not been completed and less than 30 rounds have been played, a single round of the game occurs:

1. Tell the player the current round.
2. Roll the die, and tell the player the result.
3. Whichever body part the roll corresponds to, if that body part can be used, increase the count for that body part. If the player loses a turn, let the player know.
4. Print the current caterpillar.
5. Ask the user to "Press enter to continue..." by strategically using the `input()` function to pause after each round.

The player wins if the caterpillar is complete in up to 30 rounds. If the player wins, tell the player how many rounds it took to win. Otherwise, tell the player they lost.

For this function, you should utilize the other function you have coded as much as possible. See the example output at the end of this file.

Program Requirements

Your program must be well structured and meet the following specifications:

- Stick to using variables instead of literals as much as possible. On that note...
- Use constants where appropriate. Think about which values never change through each play of the game, and use constants instead of literal values.
- Your prompts and output should look identical to those provided in this document.
- Your program must be commented appropriately, specifically you must:
  - Include an appropriate file prolog at the top of the source file.
  - Include a comment above each function including a description of the parameters, function, and return statement.
  - Include appropriate comments throughout the program.
  - Use meaningful variable names.
- Your program should not include a main function or any floating code. If you want to play the game, call it from the tester.

What to Submit

Please submit the following file to Canvas by the due date and time.

- `caterpillar.py`

Remember, all of the files must be named exactly as indicated above, with the same case and with no spaces or special characters.

Example Output

```
********************************************
**** Welcome to the game of Caterpillar ****
********************************************

         \ /
          O
* * *-* * *
| | | | | |

Roll 1

You rolled a 3!
You got a body!

      * * *

Press enter to continue...
Roll 2

You rolled a 6!
You lose a turn.

      * * *

Press enter to continue...
Roll 3

You rolled a 2!
You got a head!

          O
      * * *

Press enter to continue...
Roll 4

You rolled a 5!
You got a leg!

          O
      * * *
          |

Press enter to continue...
Roll 5

You rolled a 4!
You got a leg!

          O
      * * *
        | |

Press enter to continue...
```

```
Roll 6

You rolled a 5!
You got a leg!

          O
      * * *
      | | |

Press enter to continue...
Roll 7

You rolled a 1!
You got an antenna!

         \
          O
      * * *
      | | |

Press enter to continue...
Roll 8

You rolled a 5!
You need another body segment for this leg.

         \
          O
      * * *
      | | |

Press enter to continue...
Roll 9

You rolled a 1!
You got an antenna!

         \ /
          O
      * * *
      | | |

Press enter to continue...
Roll 10

You rolled a 6!
You lose a turn.

         \ /
          O
      * * *
      | | |

Press enter to continue...
```

```
Roll 11

You rolled a 3!
You got a body!

         \ /
          O
* * *-* * *
      | | |

Press enter to continue...
Roll 12

You rolled a 4!
You got a leg!

         \ /
          O
* * *-* * *
    | | | |

Press enter to continue...
Roll 13

You rolled a 2!
You've already got a head.

         \ /
          O
* * *-* * *
    | | | |

Press enter to continue...
Roll 14

You rolled a 6!
You lose a turn.

         \ /
          O
* * *-* * *
    | | | |

Press enter to continue...
Roll 15

You rolled a 5!
You got a leg!

         \ /
          O
* * *-* * *
  | | | | |

Press enter to continue...
```

```
Roll 16

You rolled a 5!
You got a leg!

         \ /
          O
* * *-* * *
| | | | | |

Press enter to continue...
You won in 16 rounds!
```