

Assignment 1: Dream Pet 🐱

Dream Pet 🐱

Due date 11:59 PM Sunday 5 September 2021 (Sydney local time)

Changes to the description

Periodic updates about this description may appear here. **Your program should behave exactly like the examples given.** More examples may be published. More test cases will be released gradually.

***UPDATE 6 (26/08)* Clarifications:**

- Amounts for lavish food methodically preparation are decimal values between 0 (exclusive) and 50 (inclusive).
- Feeding and no feeding output for lavish food methodically preparation:
<https://edstem.org/au/courses/6769/discussion/568523>
- More test cases will be released.
- Kitchen 2 pet 1 present pet string changed to `<name>`, do you want to call `<pet2_name>`, or pet or play with `<pet1_name>`? (Call/Play/Pet) This shouldn't have too much impact.

UPDATE 5 (24/08) The test cases output has changed so you can also see the input and new lines. Hopefully, this makes it easier for everyone. Sorry for the confusion. More test cases will be released tomorrow. Get ready!

UPDATE 4 Inputs for the 4 given full examples are provided under the `test_cases` directory in the workspace. You need to submit your code first and then reset your workspace to scaffold to see them. You can use the `<` redirect command to run these inputs.

E.g. `python3 dream_pet.py < test_cases/basic_pass.in`

UPDATE 3 Added a full example for two pets, fixing typos... more test cases to come...

UPDATE 2 Thanks to all those who are reading through the spec very carefully! There are a few little typos in some examples/spec. They are fixed now.

UPDATE 1 There was a typo in `GAME_START_BANNER` It is fixed now, please reset your workspace to scaffold after submission or use this: `GAME_START_BANNER =`


`"\n=====GAME START====="`

Restrictions

 No user-defined functions, no classes, no dictionaries

 Do not import any modules

A message to all students about posting on Ed

 This is an assignment, and staff are not permitted to give guidance on your code or how to solve the specific problem. That is the purpose of the assessment that you are required to perform to achieve the grade.

You may ask clarification questions about the assignment description. This is often necessary to implement functionality that is otherwise ambiguous.

The assignment description is not intended to be complete and you can confirm your assumptions in a form of a question. In asking the question you should be quoting the description you are asking about.

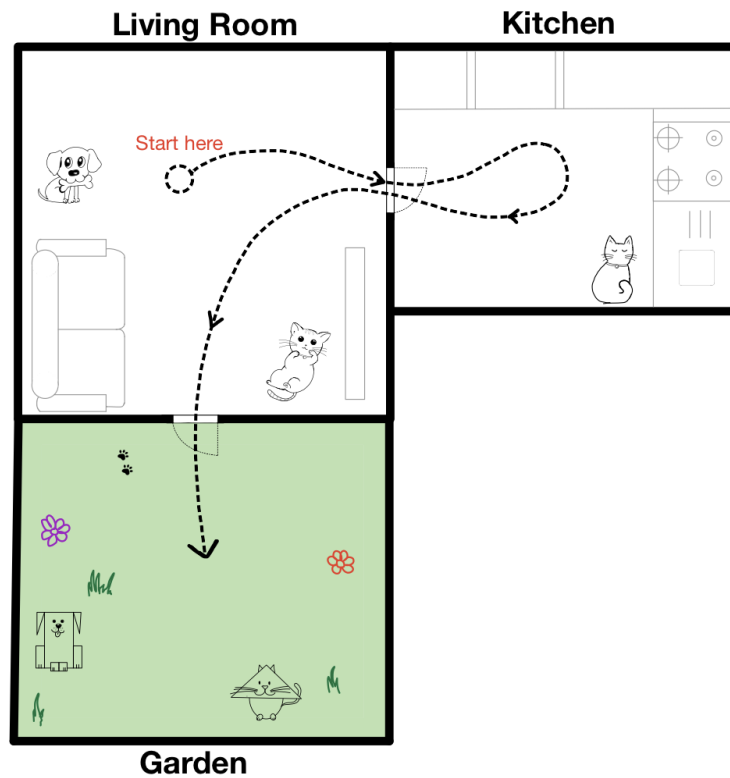
If you have a question to ask on Ed please search before asking. However, remember that you should not be posting any assignment code publicly, as this would constitute academic dishonesty. Also, do not wait too long before starting. This assignment needs time and sustained effort.

Problem description

You are writing an interactive program for your best friend, a pet lover. Unfortunately, your friend's parents prohibited them from getting a pet, so they could only retreat to the digital world and resort to writing their own pet simulation. However, their imagination exceeds their capabilities as their programming skills are somewhat lackluster. Can you please help them and write a pet simulation game?

The target program simulates the process of a player interacting with his/her pet(s) and ends when the player leaves the house. Within the house, there are pets and rooms.

- There are 3 rooms: the living room, the kitchen, and the garden. The player starts in the living room. Then, they move to the kitchen, back to the living room, and finally to the garden. They will perform different actions in each room. The game ends in the garden.



- There will be 1 or 2 pets: any combination of dogs and cats. Each pet has the following properties: a name, the room they start in, happiness level (0 - 100), and hunger level (0 - 100).
 - The values of happiness level and hunger level must stay within range throughout the game. E.g. if the happiness value increases to 110, then cap it at 100. If the hunger value decreases to -10, then bring it back to 0.
 - If at any point in the game a pet is too hungry (hunger level ≥ 90), the pet will always go to or stay in the kitchen. Calling them from another room will fail in this case.

At the start of the program, the player will be greeted and prompted to enter data for the state of their pets. The program then walks the player through the rooms in the above-mentioned order. In each room, the program will prompt the player to perform a series of actions that affect the state of the pets and the player. When the player finishes traversing the rooms, the program ends, and the state of the pets will be reported.

- The game will either end successfully or fail. The criteria are in the **Final outcome** section.

The player will be asked a series of questions to determine which steps the program should perform. All questions (except for ones in the **State data entry** section) have a fixed number of possible answers. The options will be indicated in brackets, after the question mark, with / separating each option. e.g. What type of animal is your pet? (Cat/Dog) , you should type in cat or dog as your answer. You can assume the answer given will always be one of the options, the behaviour of incorrect input is undefined. The answer should be case insensitive (except for names).

Program operation

Invoked the program by running the command

```
$ python3 dream_pet.py
```

Program data

The happy pet sound for cats is **Meow** and **Woof** for dogs. The angry pet sound for cats is **Hisssss** and **Grrrrrr** for dogs.

First message

The program begins by greeting the player with an 8 line banner and a prompt for the player's name. The banner is shown below, and the prompt should be `Welcome to Dream Pet. May I grab your name please.` After getting their name, say their name and welcome them to the game. The welcome message should be `Nice to see you <name>. I hope you enjoy our short interaction.`

Note. `<name>` is used to represent the player's input. Do NOT print `<name>` out verbatim!

i In the following examples, anything with a # in front will be user input from stdin.

Example:

[illegible]

State data entry

The next step is to get the initial values of the pets. Print the `GAME_SETUP_BANNER`. The program will ask the player for the number of pets: Now, tell me <name>, would you like one or two pets? (1/2) . For 1 pet, the following 5 questions must be asked in order:

1. What type of animal is your pet? (Cat/Dog)?
2. What is this `<pet_type>` called?
3. Where is `<pet_name>` right now? (Living Room/Kitchen/Garden)
4. What is `<pet_name>`'s hunger value?
5. What is `<pet_name>`'s happiness value?

For 2 pets, the program should ask the following 10 questions in order:

1. What type of animal is your first pet? (Cat/Dog)?
2. What is this <pet1_type> called?
3. Where is <pet1_name> right now? (Living Room/Kitchen/Garden)
4. What is <pet1_name>'s hunger value?
5. What is <pet1_name>'s happiness value?
6. What type of animal is your second pet? (Cat/Dog)?
7. What is this <pet2_type> called?
8. Where is <pet2_name> right now? (Living Room/Kitchen/Garden)
9. What is <pet2_name>'s hunger value?
10. What is <pet2_name>'s happiness value?



Note. <pet_type> is the player input for question 1. <pet_name> is the player input for question 2. Assume that the input for the hunger value and happiness value will always be floating-point values and will be in the range of 0 and 100 inclusive. Only the pet name is case-sensitive.

Example setup 1:

```
...
=====GAME SETUP=====
Now, tell me Mark, would you like one or two pets? (1/2) #1
What type of animal is your pet? (Cat/Dog)? #Dog
What is this dog called? #Rocket
Where is Rocket right now? (Living Room/Kitchen/Garden) #Garden
What is Rocket's hunger value? #50
What is Rocket's happiness value? #50
...
```

Example setup 2:

```
...
=====GAME SETUP=====
Now, tell me Mark, would you like one or two pets? (1/2) #2
What type of animal is your first pet? (Cat/Dog)? #Dog
What is this dog called? #Rocket
Where is Rocket right now? (Living Room/Kitchen/Garden) #Garden
What is Rocket's hunger value? #1
What is Rocket's happiness value? #99
What type of animal is your second pet? (Cat/Dog)? #Cat
What is this cat called? #BigOrange
Where is BigOrange right now? (Living Room/Kitchen/Garden) #Kitchen
What is BigOrange's hunger value? #0
What is BigOrange's happiness value? #100
...
```

Game Status

The game will now print out the current status of the pets. First, print `STATUS_BANNER`. Then follow by

<name>, here is a summary of the current status of your pets: .Then the game prints the status of each pet on a new line in the order of the first and second pet: Your little <pet_type>, <pet_name>, is in the <pet_room> and is <happiness_level>% happy and <hunger_level>% hungry. Round all floats to 2 decimal places and represent the values as a percentage in the summary.

The game status should be printed at the end of the **State data entry** and **Calculate the final game state** sections and after the player leaves each room.

Example 2 pets status:

```
...
=====STATUS=====
Mark, here is a summary of the current status of your pets:
Your little dog, Rocket, is in the garden and is 99.00% happy and 1.00% hungry.
Your little cat, BigOrange, is in the kitchen and is 100.00% happy and 0.00% hungry.
....
```

Game Start

Now the real simulation begins! Signal the user by printing the `GAME_START_BANNER`.

In the Living Room

Tell the player that they are in the living room by printing the `LIVING_ROOM_BANNER`. Then for any pet that is already in the living room, it should print `<pet_happy_sound>, <pet_name> is in the living room!` in the order of the first and second pet.

In the living room, the player can perform one of three actions:

1. Interact: Interact with your pets
2. Find: Find some pet food
3. Leave: Leave the living room

Initially, all three options are available to the player, the action is selected by asking `We are in the living room, which of the following actions would you like to perform? (Interact/Find/Leave)`. Given the user input, the program should behave like the following:

- **Interact:** Different interactions will be offered based on the number of pets in the game and in the living room. Please see the **Interactions with the pet** section for more details on call, pet and play.
 - The player owns 2 pets:
 - 0 pet in the living room: The player wants to interact with the pets but no pet is here! They have one chance to call in the pets. The game will print `No pet is in the living room, who do you want to call?`

(<pet1_name>/<pet2_name>/Both) If no pets came, the game will ask the player None of the pets are here, which one of the following actions would you like to perform next? (Find/Leave) If one pet came, ask <name>, do you want to pet or play with <pet_name>? (Pet/Play) If both pets came, the game should now follow the **2 pets in the living room** section.

- 1 pet in the living room: One of the pets are in the living room, the player can either try to call the other pet, or pet or play with the present pet. The game will ask <name>, do you want to call <absent_pet_name>, or pet or play with <present_pet_name>? (Call/Play/Pet) . If the player calls the absent pet and fails, the game will let them pet or play with the one present: Do you want to pet or play with <present_pet_name>? (Play/Pet) If the call succeeds, the game should now follow the **2 pets in the living room** section.
- 2 pets in the living room: The game will say Both <pet1_name> and <pet2_name> are in the living room! Do you want to pet or play with them? (Pet/Play) The player will enter their choice and the program will follow up with the question <name>, your choice is <choice>, who do you want to interact with? (<pet1_name>/<pet2_name>/Both)

◦ The player owns 1 pet:

- Pet not in the living room: The player wants to interact with the pet but it is not here! The game will print <pet_name> is not in the living room, do you want to call it? (Yes/No) If the player does not want to call the pet or if the call fails, the program will ask the player <name>, which one of the following actions would you like to perform next? (Find/Leave) If the call succeeds, the game should now follow the **Pet in the living room** section.
- Pet in the living room: The game will ask the player <name>, do you want to pet or play with <pet_name>? (Pet/Play)

✓ After the player either pets or plays with a pet, the program will ask <name>, which one of the following actions would you like to perform next? (Find/Leave)

- **Find**: The player finds pet food. There are two types of pet food: simple pet food and lavish pet food. Ask the player which type of pet food they would like to look for Which type of pet food do you want to look for? (Simple/Lavish)
 - If the player chooses the *simple* pet food, tell them it was found in the overhead cupboard: Looking around the room, you found the simple pet food stored in the corner of the overhead cupboard.
 - The *lavish* pet food was stored in the delivery box, tell them: You remembered that the lavish pet food was delivered yesterday, and you grabbed it out of the box.
 - After finding the food, the player leaves the living room and enters the kitchen. Say Great, now we have some pet food, lets go to the kitchen!

- **Leave:** The game will say `Alright, lets move on to the kitchen immediately!` Then the current status of the game will be printed. The player leaves the living room and enters the kitchen.

Generate a summary of the pet status.

In the Kitchen

The player moves to the Kitchen after finishing their actions in the Living Room. Again, print the `KITCHEN_BANNER` first. Then for any pet that is already in the kitchen, it should print `<pet_happy_sound>`, `<pet_name>` is already in the kitchen waiting for you! in the order of the first and second pet.

In the Kitchen, the player chooses one of the following three actions to perform:

1. Interact: Interact with your pets
2. Prepare: Prepare some pet food
3. Leave: The player leaves the kitchen

The program gets a response by asking `Now that we are in the kitchen, what would you like to do? (Interact/Prepare/Leave)`. Given the player's input, one of the following three paths can be picked.

- **Interact:** Different interactions will be offered based on the number of pets in the game and in the living room. Please see the **Interactions with the pet** section for more details on call, pet and play.
 - The player owns 2 pets:
 - 0 pet in the kitchen: The player wants to interact with the pets but no pet is here! They have one chance to call in the pets. The game will print `No pet is in the kitchen, who do you want to call? (<pet1_name>/<pet2_name>/Both)` If no pets came, the game will ask the player `None of the pets are here, which one of the following actions would you like to perform next? (Prepare/Leave)` If one pet came, ask `Do you want to pet or play with <pet_name>? (Play/Pet)` If both pets came, the game should now follow the **"2 pets in the kitchen"** section.
 - 1 pet in the kitchen: One of the pets are in the kitchen, the player can either try to call the other pet, or pet or play with the present pet. The game will ask `<name>, do you want to call <pet2_name>, or pet or play with <pet1_name>? (Call/Play/Pet)` If the player calls the other pet and fails, the game will let them pet or play with the one present `Do you want to pet or play with <pet1_name>? (Play/Pet)` If the call succeeds, the game should now follow the **"2 pets in the kitchen"** section.

- 2 pets in the kitchen: The game will say Both <pet1_name> and <pet2_name> are in the kitchen! Do you want to pet or play with them? (Pet/Play) The player will enter their choice and the program will follow up with the question <name>, your choice is <choice>, who do you want to interact with? (<pet1_name>/<pet2_name>/Both)

◦ The player owns 1 pet:

- Pet not in the kitchen: The player wants to interact with the pet but it is not here! The game will print <pet_name> is not in the kitchen, do you want to call it? (Yes/No) If the player does not want to call the pet or if the call fails, the program will ask the player Which one of the following actions would you like to perform next? (Prepare/Leave) If the call succeeds, the game should now follow the "**Pet in the kitchen**" section.
- Pet in the kitchen: The game will ask the player <name>, do you want to pet or play with <pet_name>? (Pet/Play)



After the player either pets or plays with a pet, the program will ask <name>, which one of the following actions would you like to perform next? (Prepare/Leave)

- **Prepare**: The player prepares the pet food. If the player did not find the pet food in the previous room, then print out You forgot to find where the pet food is, so there appears to be nothing you can prepare. Let's leave. Otherwise, present the player with two ways to prepare the food. One way is to prepare the food in a methodical way, while the other is to prepare it in a chaotic fashion. Prompt the player to select the methodology: With the pet food in front of you, how would you like to prepare it? (Methodical/Chaotic).
 - If methodical preparation was selected and the lavish pet food was found, then refer to the "**Methodically preparing lavish pet food**" section. For both lavish and simple pet food, print You carefully spend 1 hour making the perfect meal. The end result looks like something edible for humans. after completion.
 - If chaotic preparation was selected, then the result depends on the type of pet food found. If the simple pet food was found, then print The blender blends the simple pet food into a paste that looks a bit better than what you started with. If the lavish pet food was found, print The complex design of the original recipe meant that in such a haphazard preparation ruined the taste and texture. This ruins the meal, and if it was fed to the pet then the pet will become unhappy.
 - After preparing the food, all pets in other rooms will come into the kitchen without needing to call them. The game will print <pet_happy_sound>, <pet_name> heard you preparing the food, it dashed into the kitchen! for the first and second pet in order.
 - The player now has the option to feed the pet or just leave the kitchen. This is prompted by You look at the prepared food in your hands. What do you feel like doing

now? (Feed/Leave) Please see the **Interactions with the pet** section for more details on feed.

- If the player left the kitchen without feeding the pet, then print `Forget about the prepared pet food, let's leave the kitchen.` If the pet was fed, then print `The <pet_type> finished the feast, and you leave the kitchen.` If the pet is not fed, print `Okay, no feeding is needed, you go to the living room.`
- **Leave:** The player leaves the kitchen and returns to the living room. Print `Alright, lets go back to the living room immediately!`

Generate a summary of the pet status.

In the Living Room (again)

Once again we are back in the living room! Any pets that are in the kitchen will follow you to the living room if their happiness level is greater than 60 and their hunger level is less than 50 after feeding.

Again, print the `LIVING_ROOM_BANNER` first. Then for any pet that is in the living room, it should print `<pet_happy_sound>`, `<pet_name> is in the living room!` in the order of the first and second pet.

We present the player with possible actions based on the number of owned pets and the number of pets that are currently in the living room.

- The player owns 2 pets:
 - **0 pets in the living room:** the program should print `No pets are in the living room :(. Would you like to Call or Leave? (Call/Leave)` . If the player chooses to call the pets we ask the player which pets they would like to call by prompting `Who are we calling? (<pet1_name>/<pet2_name>/Both)` . If no pets came, or if the player wants to leave, the game will inform the player `None of the pets are here, lets finish this game!` If one pet came, ask `Do you want to say goodbye to <pet_name> or put it in the cage? (Goodbye/Cage/Leave)` If both pets came, the game should now follow the **2 pets in the living room** section below.
 - **1 pet in the living room:** the program should ask the player `What would you like to do now? (Call/Goodbye/Cage/Leave)` If the player calls the other pet and fails, the game will let them cage or say goodbye to the one present `Do you want to say goodbye to <pet_name>, put it in the cage, or leave? (Goodbye/Cage/Leave)` If the call succeeds, the game should now follow the **"2 pets in the living room"** section below.
 - **2 pets in the living room:** The game will say `Both <pet1_name> and <pet2_name> are in the living room! Do you want to say goodbye, put them in the cage, or leave? (Goodbye/Cage/Leave)` If the player chooses `Goodbye` or `Cage`, the action should be applied to both pets in the room.
- The player owns 1 pet:
 - **Pet not in the living room:** the program should print `<pet_name> is not in the`

living room, do you want to call it? (Yes/No) . If the player chooses No or the pet fails to come to the living room, the program should print Well, I guess you can leave then. , and the player goes to the Garden. Otherwise, follow the point below.

- **Pet in the living room:** the program should prompt the player to choose the next action by asking what would you like to do now? (Goodbye/Cage/Leave) . Please see the **Interactions with the pet** section for the expected behaviour for Goodbye and Cage.

✓ Only print okie, lets finish this game! if the player inputs leave or either says goodbye or cages the pet and goes to the garden.

Generate a summary of the pet status.

Calculate the final game state

We are now finally in the garden and ready to leave the house! The game is coming to an end. Print the `GARDEN_BANNER` with the message `Good job <name>, now we are in the garden, let's see how you are performing.`

Example:

• • •

```

-----
|  _ _ \          | |
| |  \ /  _ _ _ _ _ _ _ _ | |  _ _ _ _ _ _
| |  _ _ / _ ` || ' _ _ | / _ ` | / _ \ | ' _ \
| | _ \ \ | ( _ | | | | | ( _ | | | _ _ / | | | |
 \ _ _ _ / \ _ _ , _ | _ | \ _ _ , _ | \ _ _ | | _ | _ |

```

Good job Mark, now we are in the garden, let's see how you are performing.

• • •

The game will now perform some calculations based on how much time the player spent with each pet. **Time spent** with the pets will be represented by the number of successful interactions the player engaged in with each pet. The number of interactions is detailed in the **Interactions with the pet** section.

We adjust the final happiness value for each pet based on the following formula. Let us define 2 mathematical functions first:

1. $f(\text{happiness level}, \text{time spent with pet}) = \text{happiness level} + \text{time spent with pet} * 1$

$$2. g(hunger\ level) = \begin{cases} -1, & \text{if } hunger\ level - 50 < 0 \\ 1, & \text{if } hunger\ level - 50 > 0 \\ 0, & \text{Otherwise} \end{cases}$$

The following equation for the final happiness level is thus derived. If the hunger level of the pet is

more than 50, the happiness value will decrease. If the hunger level of the pet is less than 50, the happiness value will increase. Otherwise, the happiness value will not change. The more time the player spends with the pet, the happier the pet will be (caging the pet should not count towards time spent).

$$final\ happiness\ level = f(happiness\ level, time\ spent\ with\ pet) - g(hunger\ level) * hun$$

Remember happiness value should stay within the range of 0-100. Generate a summary of the pet status.

End of the program

Print the `GAME_END_BANNER`.

The game will either end successfully or fail. The criteria for success is all pets are happy (happiness level ≥ 80) and not hungry (hunger level ≤ 20).

If the outcome was a success, print `Congratulations! You did an amazing job looking after your pets! Hope to see you again soon!` Otherwise, print `Unfortunately! You didn't do a good job looking after your pets! Hopefully you will do better next time.`

Always print `Bye <name>!` at the end of the program.

Example success:

```
....
=====GAME END=====
Congratulations! You did an amazing job looking after your pets! Hope to see you again soon!
Bye Mark!
```

Example failure:

```
....
=====GAME END=====
Unfortunately! You didn't do a good job looking after your pets! Hopefully you will do better next
time.
Bye Mark!
```

Interactions with the pet

The player can interact with each pet in the six following ways. If the player chose to perform an action on BOTH of the pets, then print the results in the order of the first pet and then the second pet.

- **Call:** The player tries to call the pet over. However, the pet will only come to the player's room if it is happy (happiness level ≥ 20) and it is not too hungry (hunger level < 90). If it is too hungry

it will stay in the kitchen). After the pet reaches the current room, the player can further interact with the pet.

- Success: If the pet came, the game will print `<pet_happy_sound>`, `<pet_name> came from the <original_room>`, it is with you in the `<current_room>` now!
- Fail: If the pet does not come, the game should print `<pet_angry_sound>! Oh no, <pet_name> is a bit angry, it doesn't want to come!`
- **Pet:** Petting the pet. If the pet is a dog, print `<pet_happy_sound>`, `<pet_name> is really enjoying that rub in the tummy!!`. If the pet is a cat, print `<pet_happy_sound>`, `<pet_name> loves that good scratch under the chin, it is purring so loud! Such a purr machine!!` This will increase the pet's happiness by 9.26.
- **Play:** Playing with the pet. If the pet is a dog, print `<pet_happy_sound>`, `<pet_name> is chasing after that ball! It's so excited it almost knocked over the table!!`. If the pet is a cat, print `<pet_happy_sound>`, `<pet_name> loves pouncing at that cat wand! Such a predator!` This will increase both the pet's happiness level by 13.21 and the hunger level by 10.90.
- **Feed:** Feeding the pet. If the player owns 2 pets, the game will ask the player `Which pet do you want to feed? (<pet1_name>/<pet2_name>)`, else the player will feed the only pet. If the pet is already full (hunger level = 0) don't feed the pet, print `<pet_name> is already full, you don't need to feed it.` In this case, if there is another pet in the kitchen, ask `Would you like to feed <other_pet> instead? (Yes/No)` If the player tries to feed the other pet and it is also full, print `<other_name> is already full, you don't need to feed it.`
 - For simple pet food, there will always be 100 grams of the final product after preparation. You feed 20 grams of the food at a time and the pet's hunger value will decrease by 1 per 2 grams of the food. The feeding either stops when the pet is full (hunger level 0) or when the food runs out. Each time the player feeds the pet the game will print `You fed <pet_name> <amount> grams of food, it is now <hunger_level>% hungry.` If the pet gets full while eating the 20 grams of the food, still feed the pet 20 grams. If the pet becomes full, print `<pet_happy_sound>`, `<pet_name> is satiated from the feast!` If the food runs out, print `There's no more prepared food left!`. If both occur, then print the two messages in the order above.
 - For lavish pet food prepared chaotically, it will be the same feeding procedure as simple pet food, however, always remember to decrease the happiness of the pet by 10 after feeding as its lavish food is ruined!
 - For lavish pet food prepared methodically, see the section **Methodically preparing lavish pet food** for the feeding instructions.
- **Goodbye:** Say goodbye to the pet. The game will output `You gently pat <pet_name>'s head and whispers 'see you soon, little one' in its ear, <pet_name> replies: "<pet_happy_sound>"!` This will increase the pet's happiness by 10.
- **Cage:** place the pet in a cage. The player wants to make sure their pet is not naughty while they

are away, so they cage the poor little animal. The pet cries in despair, and the game will say `You put <pet_name> in its cage, who "<pet_angry_sound>" in protest.` . This will decrease the pet's happiness by 20.

Methodically preparing lavish pet food

If the player chooses to prepare the lavish pet food found earlier in a methodical way, the player needs to decide on the portions of water and dry food to make the meal for their pet(s). This is done by prompting the player to `Enter the amount of water:` and `Enter the amount of dry food:` . Both amounts are decimal values between 0 (exclusive) and 50 (inclusive). Unfortunately, though, the only available utensils are:

- a mini measuring cup with 15ml capacity, and
- a teaspoon that fits at most 7g of the dry pet food.

So, once the player decides on the amounts, the program prints `Added <x> cups to the bowl` after each added cup and `Added <y> spoons to the bowl` after each added teaspoon, where `<x>` is the cup number, and `<y>` is the spoon number, incremented accordingly. `x` and `y` should always be integers. Refer to the **Full examples** section below for example input and the expected corresponding outputs.

After preparing the food, if the player chooses to feed their pet, the program will calculate the effect on the pet, keeping in mind that the pet is very picky when it comes to his/her food.

To calculate, you need to take the [absolute difference](#) between the entered amounts of water and dry food and round it to the nearest integer. Then,

- If the difference is 0, then we consider this as the perfect ratio and will increase the pet's happiness level by 20 and decrease the hunger level by 20, leaving the pet happy and full.
- If the absolute difference is not 0 but is less than 15, the effect will be as follows:
 - if there is more water than dry food, the pet will be happy but not full, increase the pet's happiness by 20 and decrease the hunger by 10.
 - On the other hand, if there is more dry food than water, the pet will be full but not too satisfied, increase the pet's happiness by 10 and decrease the hunger by 20.
- Otherwise, if the absolute difference is an odd number, the pet will refuse to eat the food and will become really grumpy afterwards, therefore we decrease the happiness level and increase the hunger level by 20. If it's even however, the pet will reluctantly eat some of the food. Meaning, happiness level will decrease by 15, and hunger will decrease by 10.

Example:

...

```
'''preparing lavish food found in the living room'''
```

With the pet food in front of you, how would you like to prepare it? (Methodical/Chaotic) #methodical

Enter the amount of water: #32.75

Enter the amount of dry food: #5.3

Added 1 cups to the bowl

Added 2 cups to the bowl

Added 3 cups to the bowl

Added 1 spoons to the bowl

...

Tips and error cases to consider

- ~~You should always print an empty line directly before asking the player for an answer.~~
- Answers to questions are case-insensitive. For example,
- when answering a `Y/N` question, `Y` is valid, and `y` is valid.
- To print a string containing single quotes, enclose it in double-quotes, and vice versa (or put a backslash before it).
- If the user input does not match a valid option or is not within a valid range for numerical inputs, the behaviour is undefined.
- Use input redirection.

Formatting the output

Alignment of text

Notice that the output of these examples aligns. To do this, you should consider using the `ljust` and `rjust` string functions.

Decimal places

There is a difference between rounding the floating-point values to store in memory and rounding the floating-point value to print to screen:

```
# modify memory as rounded value
value = 4.854
value = round(value, 2)
print(value)

# print the rounded value
value = 4.854
print(round(value, 2))
```

All calculations round to 2 decimal places only when printing.

$\bar{1}$ $\bar{2}$ $\bar{3}$ $\bar{4}$ $\bar{5}$ $\bar{6}$ $\bar{7}$ $\bar{8}$ $\bar{9}$ $\bar{0}$
 | | / / (_) | | | |
 | | / / - | | - - - - | | - - - - - - -


```

      \ | || __| / __|| ' _ \ / _ \ | ' _ \
      | \ \ | || | _ | ( __ | | | || __/ | | |
      \_ | \_/ | _ | \__ | \__|| _ | | _ | \__|| _ | | _ |

```

Now that we are in the kitchen, what would you like to do? (Interact/Prepare/Leave) #prepare
With the pet food in front of you, how would you like to prepare it? (Methodical/Chaotic) #chaotic
The blender blends the simple pet food into a paste that looks a bit better than what you started with.

Meow, BigOrange heard you preparing the food, it dashed into the kitchen!

You look at the prepared food in your hands. What do you feel like doing now? (Feed/Leave) #feed

You fed BigOrange 20 grams of food, it is now 60.00% hungry.

You fed BigOrange 40 grams of food, it is now 50.00% hungry.

You fed BigOrange 60 grams of food, it is now 40.00% hungry.

You fed BigOrange 80 grams of food, it is now 30.00% hungry.

You fed BigOrange 100 grams of food, it is now 20.00% hungry.

There's no more prepared food left!

The cat finished the feast, and you leave the kitchen.

=====STATUS=====

Mark, here is a summary of the current status of your pets:

Your little cat, BigOrange, is in the kitchen and is 30.00% happy and 20.00% hungry.

```

      _      _      _      _
      | |      ( _ )      ( _ )      | _ _ _ \
      | |      _ _ _      _ _ _      _ _ _      | | _ / / _ _ _      _ _ _      _ _ _
      | |      | \ \ / / | ' _ \ / _ ' | |      // _ \ / _ \ | ' _ ` _ \
      | | _ _ | | \ \ / / | | | | ( _ | | | | \ \ ( _ | ( _ ) | | | | | |
      \ _ _ _ / _ | \ / | _ | _ | | _ \ _ , | \ _ | \ \ _ _ / \ _ _ / _ | _ | _ |
                                     _ _ / |
                                     | _ _ /

```

BigOrange is not in the living room, do you want to call it? (Yes/No) #no

Well, I guess you can leave then.

=====STATUS=====

Mark, here is a summary of the current status of your pets:

Your little cat, BigOrange, is in the kitchen and is 30.00% happy and 20.00% hungry.

```

      _ _ _ _      _
      | _ _ \      | |
      | | \ / _ _ _      _ _ _      | | _ _ _      _ _ _
      | | _ _ / _ ' | | ' _ _ | / _ ' | / _ \ | ' _ \
      | | \ \ | ( _ | | | | ( _ | | | | _ _ / | | | |
      \ _ _ _ / \ _ , _ | _ | \ _ , _ | \ _ _ | _ | _ |

```

Good job Mark, now we are in the garden, let's see how you are performing.

=====STATUS=====

Mark, here is a summary of the current status of your pets:

Your little cat, BigOrange, is in the kitchen and is 33.20% happy and 20.00% hungry.

=====GAME END=====

Unfortunately! You didn't do a good job looking after your pets! Hopefully you will do better next time.

Bye Mark!

Example 1 pet prepare lavish food and feed:

[illegible]

```
Welcome to Dream Pet. May I grab your name please. #Mark
Nice to see you Mark. I hope you enjoy our short interaction.
```

=====GAME SETUP=====

```
Now, tell me Mark, would you like one or two pets? (1/2) #1
What type of animal is your pet? (Cat/Dog)? #Dog
What is this dog called? #Rocket
Where is Rocket right now? (Living Room/Kitchen/Garden) #Garden
What is Rocket's hunger value? #50
What is Rocket's happiness value? #50
```

=====STATUS=====

Mark, here is a summary of the current status of your pets:
Your little dog, Rocket, is in the garden and is 50.00% happy and 50.00% hungry.

```
=====GAME START=====
```

	()	()		_ _ _ \
	_ _ _	_ _ _ - _ _	_ _ _ -	_ / / _ _ _ _ _ _ _ _ _
	\ \ / /	' _ \ / _ '		// _ \ / _ \ ' _ ' _ \
_ _ _	\ \ v /	(_		\ \ (_) (_)
\ _ _ _ _ / _	\ _ /	_ _	_ \ ,	\ \ \ _ _ _ / \ _ _ _ / _ _
			_ _ /	
			_ _ _ /	

We are in the living room, which of the following actions would you like to perform? (Interact/Find/Leave) #Interact

Rocket is not in the living room, do you want to call it? (Yes/No) #Yes

Woof, Rocket came from the garden, it is with you in the living room now!

Mark, do you want to pet or play with Rocket? (Pet/Play) #Pet

Woof, Rocket is really enjoying that rub in the tummy!!

Mark, which one of the following actions would you like to perform next? (Find/Leave) #Find

Which type of pet food do you want to look for? (Simple/Lavish) #Lavish

You remembered that the lavish pet food was delivered yesterday, and you grabbed it out of the box.

Great, now we have some pet food, lets go to the kitchen!

=====STATUS=====

Mark, here is a summary of the current status of your pets:
Your little dog, Rocket, is in the living room and is 59.26% happy and 50.00% hungry.

```

      _  _  _  _      _
      | | / /(_) | |      | |
      | | / / _ | | _ _ _ | | _ _ _ _ _
      |   \ | | | _ | / _ | | ' _ \ / _ \ | ' _ \
      | \ \ \ | | | _ | ( _ | | | | _ / | | | |
      \ _ \ \ / | _ \ _ \ _ _ | | _ \ _ _ | | _ \

```

Now that we are in the kitchen, what would you like to do? (Interact/Prepare/Leave) #Interact

Rocket is not in the kitchen, do you want to call it? (Yes/No) #Yes

Woof, Rocket came from the living room, it is with you in the kitchen now!

Mark, do you want to pet or play with Rocket? (Pet/Play) #pet

Woof, Rocket is really enjoying that rub in the tummy!!

Mark, which one of the following actions would you like to perform next? (Prepare/Leave) #Prepare

With the pet food in front of you, how would you like to prepare it? (Methodical/Chaotic) #Methodical

Enter the amount of water: #30

Enter the amount of dry food: #31

Added 1 cups to the bowl

Added 2 cups to the bowl

Added 1 spoons to the bowl

Added 2 spoons to the bowl

Added 3 spoons to the bowl

Added 4 spoons to the bowl

Added 5 spoons to the bowl

You carefully spend 1 hour making the perfect meal. The end result looks like something edible for humans.

You look at the prepared food in your hands. What do you feel like doing now? (Feed/Leave) #feed

The dog finished the feast, and you leave the kitchen.

=====STATUS=====

Mark, here is a summary of the current status of your pets:

Your little dog, Rocket, is in the kitchen and is 78.52% happy and 30.00% hungry.

```

      _      _      _      _
      | |  ( _ )  ( _ )      | _ _ \
      | |      _ _ _ _ _ _ _ _ | | / / _ _ _ _ _ _ _
      | |      | \ \ / / | ' _ \ / _ ' | | // _ \ / _ \ | ' _ ` _ \
      | | _ _ | | \ \ / / | | | | ( _ | | | \ \ ( _ | ( _ ) | | | | |
      \ _ _ _ / _ \ \ / | _ | _ \ _ _ | \ _ \ \ _ _ / \ _ _ / | _ | _ |
              _ _ / |
              | _ _ /

```

Woof, Rocket is in the living room!

What would you like to do now? (Goodbye/Cage/Leave) #Goodbye

You gently pat Rocket's head and whispers 'see you soon, little one' in its ear, Rocket replies: "Woof"!

Okie, lets finish this game!

=====STATUS=====

Mark, here is a summary of the current status of your pets:

Your little dog, Rocket, is in the living room and is 88.52% happy and 30.00% hungry.

Good job Mark, now we are in the garden, let's see how you are performing.

Mark, here is a summary of the current status of your pets:

```
=====GAME  END=====
```

Bye Mark!

```
$ python3 dream_pet.py
```

```
Welcome to Dream Pet. May I grab your name please. #Abbey
Nice to see you Abbey. I hope you enjoy our short interaction.
```

Now, tell me Abbey, would you like one or two pets? (1/2) #1

What type of animal is your pet? (Cat/Dog)? #cat

What is this cat called? #Kitty

Where is Kitty right now? (Living Room/Kitchen/Garden) #kitchen

What is Kitty's hunger value? #33

What is Kitty's happiness value? #75.87

Abbey, here is a summary of the current status of your pets:

```
=====GAME START=====
```

```

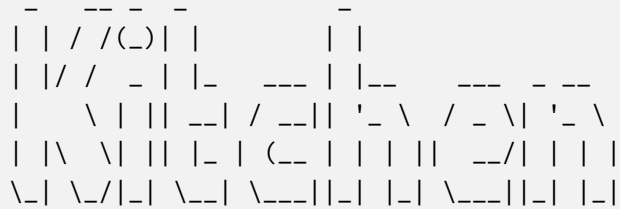
| |      (-)      (-)      | ____ \
| |      ____  ____ - ____ - | | / / ____  ____ - ____ - ____
| |      | \ \ / / | ' _ \ / _ ' | |      // _ \ / _ \ | ' _ ' _ \
| | ____ | \ \ v / | | | | | | (- | | \ \ (-) | (-) | | | | | |
\ ____ / _ \ \ / | _ | _ | _ \ __, | \ | \ \ ____ / \ ____ / _ | _ | _ |
                                     __/ |
                                     | ____/

```

We are in the living room, which of the following actions would you like to perform? (Interact/Find/Leave) #Leave
Alright, lets move on to the kitchen immediately!

=====STATUS=====

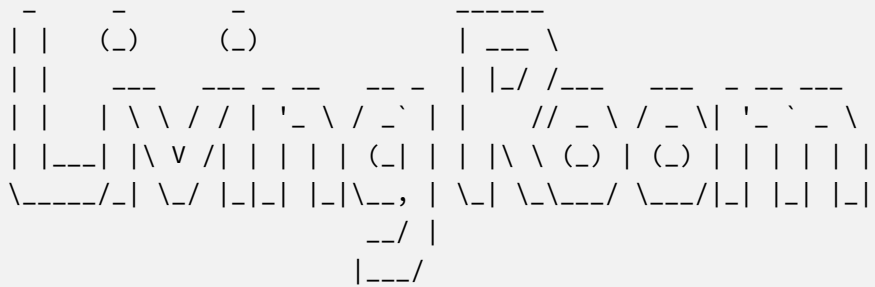
Abbey, here is a summary of the current status of your pets:
Your little cat, Kitty, is in the kitchen and is 75.87% happy and 33.00% hungry.



Meow, Kitty is already in the kitchen waiting for you!
Now that we are in the kitchen, what would you like to do? (Interact/Prepare/Leave) #Leave
Alright, lets go back to the living room immediately!

=====STATUS=====

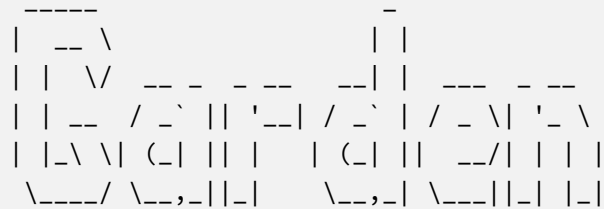
Abbey, here is a summary of the current status of your pets:
Your little cat, Kitty, is in the kitchen and is 75.87% happy and 33.00% hungry.



Kitty is not in the living room, do you want to call it? (Yes/No) #no
Well, I guess you can leave then.

=====STATUS=====

Abbey, here is a summary of the current status of your pets:
Your little cat, Kitty, is in the kitchen and is 75.87% happy and 33.00% hungry.



Good job Abbey, now we are in the garden, let's see how you are performing.

=====STATUS=====

Abbey, here is a summary of the current status of your pets:
Your little cat, Kitty, is in the kitchen and is 79.17% happy and 33.00% hungry.

=====GAME END=====

Bye Abbey!

[illegible]

=====GAME SETUP=====

=====STATUS=====

```
=====GAME START=====
```

```
Meow, Kitty is in the living room!
We are in the living room, which of the following actions would you like to perform? (Interact/Find/Leave) #interact
Amy, do you want to call Doggo, or pet or play with Kitty? (Call/Play/Pet) #pet
Meow, Kitty loves that good scratch under the chin, it is purring so loud! Such a purr machine!!
Amy, which one of the following actions would you like to perform next? (Find/Leave) #leave
Alright, lets move on to the kitchen immediately!
```

=====STATUS=====

Amy, here is a summary of the current status of your pets:

Your little cat, Kitty, is in the living room and is 53.26% happy and 30.00% hungry.

Your little dog, Doggo, is in the kitchen and is 89.00% happy and 20.00% hungry.

```

  _  _  _  _  _
  | | / / ( ) | |      | |
  | | / / _ | | _ _ _ | | _ _ _ _ _
  |   \ | | | _ | / _ | | ' _ \ / _ \ | ' _ \
  | | \ \ | | | | _ | ( _ | | | | | _ / | | | |
  \ _ | \ / | _ | \ _ | \ _ | | _ | \ _ | | _ |

```

Woof, Doggo is already in the kitchen waiting for you!

Now that we are in the kitchen, what would you like to do? (Interact/Prepare/Leave) #interact

Amy, do you want to call Kitty, or pet or play with Doggo? (Call/Play/Pet) #call

Meow, Kitty came from the living room, it is with you in the kitchen now!

Both Kitty and Doggo are in the kitchen! Do you want to pet or play with them? (Pet/Play) #play

Amy, your choice is play, who do you want to interact with? (Kitty/Doggo/Both) #both

Meow, Kitty loves pouncing at that cat wand! Such a predator!

Woof, Doggo is chasing after that ball! It's so excited it almost knocked over the table!!

Amy, which one of the following actions would you like to perform next? (Prepare/Leave) #leave

Alright, lets go back to the living room immediately!

=====STATUS=====

Amy, here is a summary of the current status of your pets:

Your little cat, Kitty, is in the kitchen and is 66.47% happy and 40.90% hungry.

Your little dog, Doggo, is in the kitchen and is 100.00% happy and 30.90% hungry.

```

  _  _  _  _  _  _  _
  | | ( ) ( )      | | _ _ \
  | | _ _ _ _ _ _ _ | | / / _ _ _ _ _
  | | | \ \ / / | ' _ \ / _ ' | | // _ \ / _ \ | ' _ ' _ \
  | | _ _ | | \ \ / / | | | | ( _ | | | \ \ ( _ | ( _ | | | | |
  \ _ _ _ / _ | \ / | _ | | _ | \ _ | \ _ \ _ _ / \ _ _ / _ | | _ |
              _ / |
              | _ _ /

```

No pets are in the living room :(. Would you like to Call or Leave? (Call/Leave) #call

Who are we calling? (Kitty/Doggo/Both) #both

Meow, Kitty came from the kitchen, it is with you in the living room now!

Woof, Doggo came from the kitchen, it is with you in the living room now!

Both Kitty and Doggo are in the living room! Do you want to say goodbye, put them in the cage, or leave? (Goodbye/Cage/Leave) #goodbye

You gently pat Kitty's head and whispers 'see you soon, little one' in its ear, Kitty replies: "Meow"!

You gently pat Doggo's head and whispers 'see you soon, little one' in its ear, Doggo replies: "Woof"!

Okie, lets finish this game!

=====STATUS=====

Amy, here is a summary of the current status of your pets:

Your little cat, Kitty, is in the living room and is 76.47% happy and 40.90% hungry.

Your little dog, Doggo, is in the living room and is 100.00% happy and 30.90% hungry.

```

      -----
      |  _ _ \      |  |
      | |  \ /  _ _ _ _ _ _ _ _ |  |  _ _ _ _ _ _
      | |  _ _ / _ ' || ' _ _ | / _ ' | / _ \ | ' _ \
      | | \ \ | ( _ | || |   | ( _ | ||   _ _ / | || |
      \ _ _ / \ _ , _ | | _   \ _ , _ | \ _ _ | | _ | |

```

Good job Amy, now we are in the garden, let's see how you are performing.

=====STATUS=====

Amy, here is a summary of the current status of your pets:
 Your little cat, Kitty, is in the living room and is 86.56% happy and 40.90% hungry.
 Your little dog, Doggo, is in the living room and is 100.00% happy and 30.90% hungry.

=====GAME END=====

Unfortunately! You didn't do a good job looking after your pets! Hopefully you will do better next time.
 Bye Amy!

Getting started

Aim for easy examples first. Things that you know are supposed to work. It is suggested that you start coding in the order of the specification, from the **First message** and **State data entry** section. Take in the required user inputs and store them in different variables for later use.

Then move to the living room for the first time. Check if there are any pets in the living room. Take in the player's action from user input and make a decision based on the input.

Code Submission

Your code submission must be made via Ed this lesson.

- To make a submission, you will need to press the "Mark" button.
- You may submit as many times as you wish without penalty.
- You are able to view previous submissions from the code submission section.
- Every submission you make includes the `README.md` and the `dream_pet.py` file

The following rules apply to your code submission:

- Only the file `dream_pet.py` will be graded by the automarker
- Only the last submission will be graded.
- Only the final `README.md` file of the last submission will be graded by your tutor
- Submission after the due date will incur a late penalty as described in the [unit of study outline](#).
- Your submission must be able to compile and run within the Ed environment provided.

After each submission, the marking system will automatically check your code against the public test

cases.

The Python version is that which is presently being used on the Ed system:

```
$ python3 --version
```

```
Python 3.9.6
```

Please ensure you carefully follow the assignment description. Your program output must exactly match the output shown in the examples.

Late is late!

The computer does not discriminate about what is late. There are two files for submission

- `dream_pet.py`
- `README.md`

They are both in your Ed workspace. Make sure these are the ones for your submission. Changes to these files after the deadline will incur a penalty for both (not individually).

Marking criteria

This assignment is 10% of your final course grade. The grade you will receive is based on the number of test cases passed as well as manual grading by your tutor.

We have provided you with some sample test cases but these do not test all the functionality described in the assignment. It is important that you thoroughly test your own code.

Automatic tests 5 / 10

These are marked by a computer. Test cases will be released gradually. Private test cases are run after due date.

- There are public test cases
- There are hidden test cases
- There are private test cases

Passing criteria: In order to obtain a pass in this assignment (at least 5/10), you will need to write a complete and correct program:

- for choosing 1 pet, either Cat or Dog, and
- for choosing any valid values of happiness/hunger/location, and
- for choosing the simple pet food, and
- for choosing the chaotic preparation of food, and
- for calculating the final game state, and

- for printing output of all the correct banners and text information of the program as described in this document.

Manual grading 5 / 10

The manual grading from your tutor will consider the style, layout and comments you provide in your code. Your tutor will not be debugging or running your code against any test cases.

- 3 marks will be allocated for questions in the `README.md` file, max 750 words in total
 - Q1: 2 marks, suggested 500 words
 - Q2: 1 mark, suggested 250 words
- 1 mark will be allocated for the style and readability of your code
- 1 mark will be allocated for comments written in your code

Edit the `README.md` file to answer these questions:

1. What were the main ideas of how you wrote your program (i.e. talk about the approach you took, any difficulties you have faced, etc.)
2. How would you change your current implementation of the program if we allowed an unlimited number of pets in the simulation and the interaction with the pets need to be in the order they entered the room you are currently in? (e.g. if pet_5 entered the room before pet_2, any interaction needs to be done with pet_5 before pet_2)

Academic declaration

By submitting this assignment you declare the following:

I declare that I have read and understood the University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy, and except where specifically acknowledged, the work contained in this assignment or project is my own work and has not been copied from other sources or been previously submitted for award or assessment.

I understand that failure to comply with the Academic Dishonesty and Plagiarism in Coursework Policy can lead to severe penalties as outlined under Chapter 8 of the University of Sydney By-Law 1999 (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

I realize that I may be asked to identify those portions of the work contributed by me and required to demonstrate my knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

I acknowledge that the School of Computer Science, in assessing this assignment, may reproduce it entirely, may provide a copy to another member of faculty, and or communicate a copy of this

assignment to a plagiarism checking service or in the house computer program, and that a copy of the assignment may be maintained by the service or the School of Computer Science for the purpose of future plagiarism checking.



Warning: Any attempts to deceive or disrupt the marking system will result in an immediate zero for the entire assignment. Negative marks can be assigned if you do not properly follow the assignment description, or your code is unnecessarily or deliberately obfuscated.