Plant Information Example

(Zylab 13.11.1)

Given a base Plant class and a derived Flower class, write a program to create a list called my-garden.

Store objects (either Plants or Flowers) in the list. Create functions:

print_list( )

↳ uses print_info( ) in each class

↳ prints all elements in the list

Read elements into the list until one inputs (-1).

Step 1: Examine the given classes.

(see PlantFlower.py)

```
class Plant:
    def __init__(self, plant_name,
                 plant_cost):
        self.plant_name = plant_name
        self.plant_cost = plant_cost
    def print_info(self):
        ⋮
```

↓

→ so, need to provide name (str), and cost (float?)

```
class Flower(Plant):
    def __init__(self, plant_name,
                 plant_cost, is_annual,
                 color_of_flowers)
```

↓

```
    def print_info(self):
```

→ so, need to provide name (str)
cost (float?)
is_annual (bool)
color_of_flowers (str)

```
from Plant Flower import Plant, Flower

my_garden = []
user_string = input()

while user_string != -1 :
    ① # check to see if plant or flower
    ② # split into variables
    ③ # store as either plant or flower
    ④ # add to the my_garden list

    user_string = input()

print_list(my_garden)
```

①/②

```
tokens = user_string.split()

type = tokens[0]
name = tokens[1]
cost = tokens[2]

if type == "plant":
    my_plant = Plant(name, cost)
    my_garden.append(my_plant)
```

③/④

```
elif type == "flower":
    is_annual = tokens[3]
    color = tokens[4]
    my_flower = Flower(name, cost,
                       is_annual, color)
    my_garden.append(my_flower)
```

```
def print_list(my_garden):
    for i in range(len(my_garden)):
        print(f'{Plant {i+1} Information:')
        my_garden[i].print_info()
        print()
```