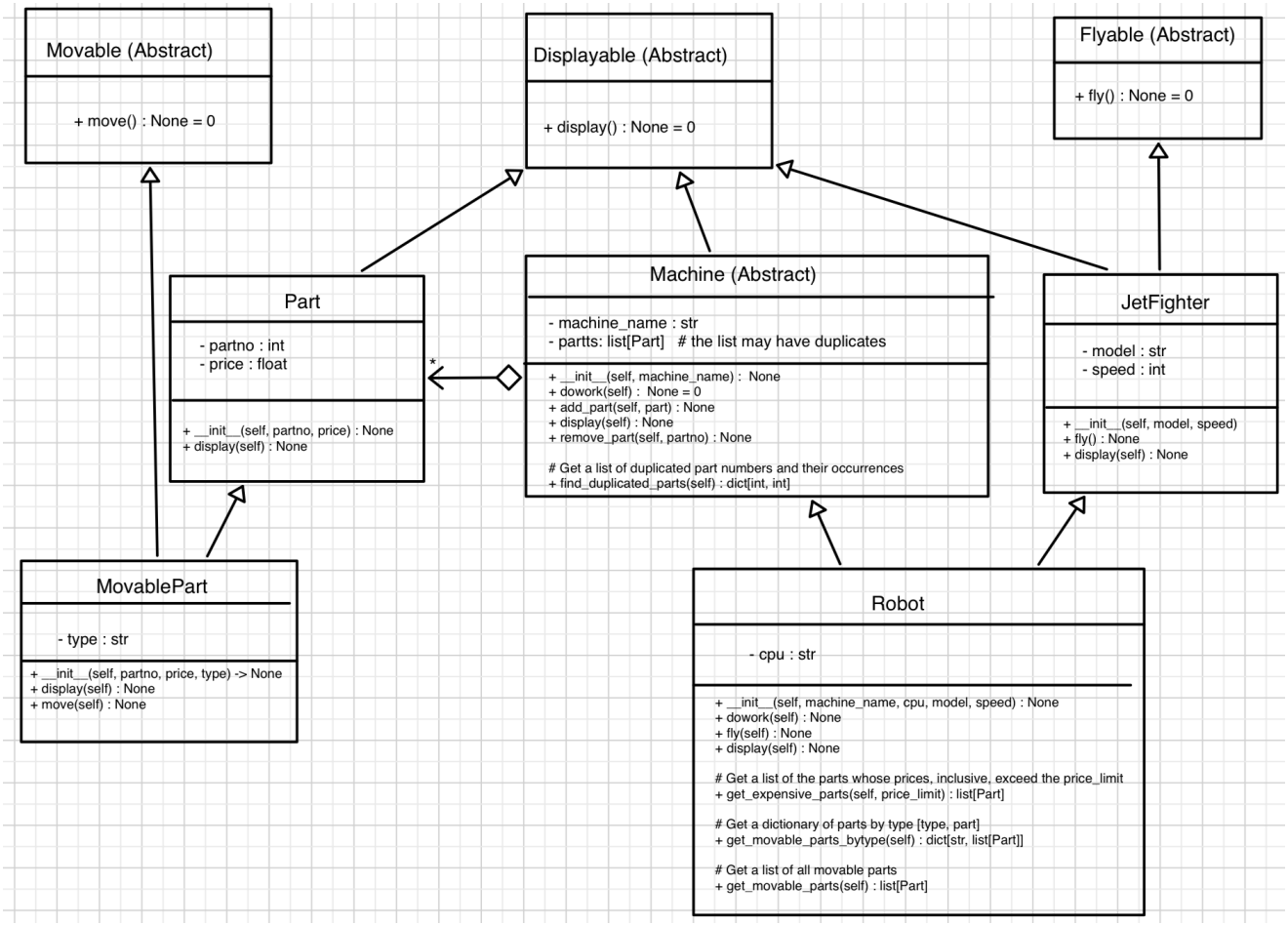
Question

Implement the following class diagrams and write a main class to test all these classes.

- Each class implements the display method of the Displayable abstract class and the method should display all the current object's and its superclass object's information. Keep in mind, you should avoid code redundancy.
- Define appropriate __init__ method so that all the objects can be created properly.
- Define appropriate getters and setters using @property if needed for other classes' methods. However, do not add any public or protected property or public get method for the private attribute 'parts' in the Machine class.



The main method for unit testing.

```
def main():
    robo = Robot('MTX', 'M1X', 'F-16', 10000)
    robo.add_part(Part(111, 100))
    robo.add_part(Part(222, 200))
    robo.add_part(Part(333, 300))
    robo.add_part(Part(222, 300))
    robo.add_part(MovablePart(555, 300, "TypeA"))
    robo.add_part(Part(111, 100))
    robo.add_part(Part(111, 100))
    robo.add_part(MovablePart(777, 300, "TypeB"))
    robo.add_part(MovablePart(655, 300, "TypeA"))
    robo.add_part(MovablePart(755, 300, "TypeA"))
    robo.add_part(MovablePart(977, 300, "TypeB"))
```

```
robo.display()
print()

print("\nRobot test flight----")
robo.fly()

print("\nRobot dowork() test ----")
robo.dowork()

print("\nDuplicated part list----")
partfreq = robo.find_duplicated_parts()
for partno in partfreq.keys():
	print(partno,'=>', partfreq[partno], 'times')

print("\nExpensive part list----")
expensive_parts = robo.get_expensive_parts(200)
for part in expensive_parts:
	part.display()

print("\nMovable part list----")
movable_parts = robo.get_movable_parts_bytype()
for type, parts in movable_parts.items():
	print("type =", type)
	for part in parts:
		part.display()
	print()

print("\nAsk movable to move----")
movable_parts = robo.get_movable_parts()
for part in movable_parts:
	part.move()

print("\nTest remove_part() ----")
robo.remove_part(333)
for part in robo:
	if part.partno == 333:
		print('Found 333')
		break;
```

The Expected output

```
cpu = M1X
machine_name = MTX
The machine has these parts:
partno = 111
price = 100

partno = 222
price = 200

partno = 333
price = 300

partno = 222
price = 300

partno = 555
price = 300
type = TypeA

partno = 111
price = 100

partno = 111
```

price = 100

partno = 777
price = 300
type = TypeB

partno = 655
price = 300
type = TypeA

partno = 755
price = 300
type = TypeA

partno = 977
price = 300
type = TypeB

model = F-16
speed = 10000

Robot test flight----
The JetFigher F-16 is flying in the sky!
The Robot MTX is flying over the ocean!

Robot doWork() test ----
The Robot MTX is assembling a big truck.

Duplicated part list----
111 => 3 times
222 => 2 times

Expensive part list----
partno = 222
price = 200
partno = 333
price = 300
partno = 222
price = 300
partno = 555
price = 300
type = TypeA
partno = 777
price = 300
type = TypeB
partno = 655
price = 300
type = TypeA
partno = 755
price = 300
type = TypeA
partno = 977
price = 300
type = TypeB

Movable part list----
type = TypeA
partno = 555
price = 300
type = TypeA
partno = 655
price = 300
type = TypeA
partno = 755
price = 300
type = TypeA

type = TypeB
partno = 777
price = 300
type = TypeB
partno = 977
price = 300
type = TypeB

Ask movable to move----
partno: 555 is moving fast!
partno: 777 is moving fast!
partno: 655 is moving fast!
partno: 755 is moving fast!
partno: 977 is moving fast!

Test remove_part() ----