Name: _____ Andrew Id: _____

15-112 Spring 2025 Quiz 10

Up to 25 minutes. No phones, no calculators, no notes, no books, no computers. Show your work!
Do not use try/except in this quiz.

1. (8 points) **Code Tracing**: Indicate what the following program prints. Place your answer (and nothing else) in the box next to the code. Ensure lists are enclosed with brackets.

```
class A:
    def __init__(self, msg, code):
        self.msg = msg
        self.code = code
        print("A")
    def __repr__(self):
        return f"A({self.msg})"
    def bar(self):
        return self.msg.replace('i',self.code )
    def __eq__(self, other):
        if isinstance(other, A):
            return self.msg.lower() == other.msg.lower()
        return False

class B(A):
    def __init__(self, x):
        super().__init__("q" + x, "42")
    def __repr__(self):
        return f"B({self.msg})"

class C:
    def __init__(self, x):
        self.x = x
    def bar(self):
        return "quiz" + self.x
    def __repr__(self):
        return f"C({self.x})"

a1 = A("quiz", "%")
a2 = A("QUIZ", "#")
b = B("uiz")
c = C("quiz")

print(a1 == a2)
print(a1 == b)
print(a1 == c)

print(type(a1) == type(b))
print(type(a1) == type(c))

print(isinstance(a2, type(a1)))
print(isinstance(b, type(a1)))
print(isinstance(c, type(a1)))

print(a1, a1.bar())
print(a2, a2.bar())
print(b, b.bar())
print(c, c.bar())
```

2. (12 points) **Free Response: Vehicles and Cars**

Write the classes `Vehicle`, and `Car` to pass the test cases shown below. Do not hardcode against the test case values, though you can assume the test cases and comments cover the needed functionality. You must use inheritance appropriately for full credit and avoid unnecessary code duplication between classes.

```
# A Vehicle has a name
v1 = Vehicle("Millennium Falcon")

# A Vehicle is initially stopped
assert(str(v1) == "Millennium Falcon: stopped")

# A vehicle can move and brake
v1.move()
assert(str(v1) == "Millennium Falcon: moving")

v1.brake()
assert(str(v1) == "Millennium Falcon: stopped")


# A Car is a vehicle that has brand name and model
c1 = Car("Nissan", "Tiida")
assert(str(c1) == "Nissan Tiida: stopped")
c1.move()

assert(str(c1) == "Nissan Tiida: moving")
c1.brake()

assert(str(c1) == "Nissan Tiida: stopped")


c2 = Car("Toyota", "Land Cruiser")
assert(str(c2) == "Toyota Land Cruiser: stopped")

# A car can tow another car
assert(c2.tow(c1) == True)

assert(str(c1) == "Nissan Tiida: stopped")
assert(str(c2) == "Toyota Land Cruiser: stopped")

# when the towing car moves, the towed car also moves
c2.move()
assert(str(c1) == "Nissan Tiida: moving")
assert(str(c2) == "Toyota Land Cruiser: moving")

# the brakes don't work when the car is being towed
c1.brake()
assert(str(c1) == "Nissan Tiida: moving") # still moving

# when the towing car brakes, both cars stop
c2.brake()
assert(str(c1) == "Nissan Tiida: stopped")
assert(str(c2) == "Toyota Land Cruiser: stopped")

# Cars can only tow other cars
c3 = Car("Nissan", "Patrol")
assert(c3.tow(v1) == False) # we cannot tow the Millenium Falcon (it's not a car)
```

Solution Space for Answer to Question 2