Name: Revanth
Mis: 112315133

1.

```
class circle:
    def __init__(self,r):
        self.radius=r
    def area(self):
        return 3.14*self.radius**2

c1=circle(5)
print(c1.area())
```

2.

```
class rectangle:
    def __init__(self,l,b):
        self.length=l
        self.width=b
    def area(self):
        return self.length*self.width
r1=rectangle(2,2)
print(r1.area())
```

3.

```
class hi:
    def __init__(self,end):
        self.end=end
    def iteration(self):
        for i in range(self.end):
            if(i%7==0):
                yield i

d=hi(15)
for i in d.iteration():
    print(i)
```

4.

```
class shape:
```

```
    def __init__(self,a=0):
        self.a=a
    def area(self):
        print(f'area of shape is {self.a}')
class square(shape):
    def __init__(self,a):
        self.length=a
    def area(self):
        print(f'area of sq is {self.length*self.length}')

s1=shape()
s1.area()
s2=square(5)
s2.area()
```

5.

```
class string:
    def getstring(self):
        self.string=input('enter a string')
    def capstring(self):
        return self.string.upper()

s1=string()
s1.getstring()
print(s1.capstring())
```

6.

```
class person:
    def getgender(self):
        print('person has no gender')
class male(person):
    def getgender(self):
        print('Male')
class female(person):
    def getgender(self):
        print('Female')

f1=female()
f1.getgender()
```

7.

```
class Package:
    def __init__(self, package_id, destination, weight, status):
        self.package_id = package_id
        self.destination = destination
        self.weight = weight
        self.status = status

    def deliver(self):
        self.status = "delivered"


class Vehicle:
    def __init__(self, vehicle_id, capacity, current_packages):
        self.vehicle_id = vehicle_id
        self.capacity = capacity
        self.current_packages = current_packages

    def load_package(self, pack_obj):
        total_weight = sum([i.weight for i in self.current_packages])
        if total_weight + pack_obj.weight <= self.capacity:
            self.current_packages.append(pack_obj)
        else:
            print("Vehicle is full")

    def deliver_packages(self):
        for i in self.current_packages:
            i.deliver()


class Truck(Vehicle):
    def deliver_packages(self):
        print("Vehicle Type: Truck")
        deli_count = 0
        for i in self.current_packages:
            if i.status == "delivered":
                deli_count += 1
        print(f"No. of packages delivered: {deli_count}")
```

```python
class Drone(Vehicle):
    def load_package(self):
        for i in self.current_packages:
            if i.weight > 5:
                raise ValueError("This package is more than 5kg")

    def deliver_packages(self):
        print("Vehicle Type: Drone")
        deli_count = 0
        for i in self.current_packages:
            if i.status == "delivered":
                deli_count += 1
        print(f"No. of packages delivered: {deli_count}")


class DeliverySystem:
    def assign_vehicle(self, veh_obj, list_pack):
        self.veh_obj = veh_obj
        self.veh_obj.current_packages = list_pack

    def dispatch(self, veh_obj):
        self.veh_obj = veh_obj
        for i in self.veh_obj.current_packages:
            yield i.deliver()


def main():
    # Changed package names
    pack_list1 = [Package("PKG1", "Mumbai", 2, "pending"), Package("PKG2",
"Chennai", 3, "intransit"), Package("PKG3", "Pune", 4, "intransit")]
    pack_list2 = [Package("PKG4", "Mumbai", 2, "pending"), Package("PKG5",
"Chennai", 3, "pending"), Package("PKG6", "Pune", 4, "pending")]

    tru_obj = Truck("V1-Truck", 1000, pack_list1)
    dron_obj = Drone("V2-Drone", 10, pack_list2)

    deli_obj = DeliverySystem()
    for i in deli_obj.dispatch(tru_obj):
        pass
```

```
    for i in deli_obj.dispatch(dron_obj):
        pass

    for i in pack_list1 + pack_list2:
        print(f"Package -> {i.package_id} Status: {i.status}")


main()
```

8.

```
from datetime import *
from random import *

count=0
def sensor_data():
    while True:
        yield (f"{datetime.now()}",randint(-10,50))

def filter_by_threshold():
    for i in sensor_data():
        if i[1]>=0 and i[1]<=40:
            yield i

mylist=[]
def smooth_temperature():
    for i in filter_by_threshold():
        mylist.append(i[1])
        if(len(mylist)<3):
           smoothed_temperature=sum(mylist)/len(mylist)
        else:
           smoothed_temperature=(mylist[-1]+mylist[-2]+mylist[-3])/3
        yield (f"{datetime.now()}",smoothed_temperature)

def convert_to_farenheit():
    for i in smooth_temperature():
        temperature_farenheit=(1.4*i[1])+32
        yield (f"{datetime.now()}",temperature_farenheit)

def main():
```

```
    for i in convert_to_farenheit():
        print(i)
        global count
        count=count+1
        if(count==20):
            break

main()
```