(b) Put the above function in a module and use another script to import the module and call the function in the script.

In [ ]:
```
from EncodeQR import EncodeQR

p_name = "Hi"
encoded_qr_list = EncodeQR(p_name)
print(encoded_qr_list)
```

```
10000000001001000100000000001101001
```

P4:Demonstrate use of object- oriented programming concepts

You are crazy about your selected domain. Today is The Day you can write a python program on YOUR DOMAIN!!!

Create a class called “Your Domain Name” which has five attributes(2 string, 3 number). Initialize the variables in the “__init_" method and write at least three more methods. With

respect to your domain, you can create methods for getting Input, update the values, verify the condition, and etc,

1. Create two objects and call the functions.
2. Implement the concept of Inheritance and Polymorphism.

In [ ]:
```
class VehicleParkingManagementSystem:
    def __init__(self, location, manager, capacity, occupied_spots, available_spots
        self.location = location #String
        self.manager = manager #String
        self.capacity = capacity #Value
        self.occupied_spots = occupied_spots #Value
        self.available_spots = available_spots #Value

    def display_info(self):
        print("Parking Location:", self.location)
        print("Manager:", self.manager)
        print("Total Capacity:", self.capacity)
        print("Occupied Spots:", self.occupied_spots)
        print("Available Spots:", self.available_spots)

    def park_vehicle(self):
        if self.available_spots > 0:
            self.available_spots -= 1
            self.occupied_spots += 1
            print("Vehicle parked successfully.")
        else:
            print("Parking is full. No available spots.")

    def vacate_spot(self):
        if self.occupied_spots > 0:
            self.available_spots += 1
            self.occupied_spots -= 1
```

```
            print("Spot vacated successfully.")
        else:
            print("No vehicles to vacate.")
```

Inheritance and Polymorphism

In [ ]:
```
class ValetParkingSystem(VehicleParkingManagementSystem):
    def __init__(self, location, manager, capacity, occupied_spots, available_spots
        super().__init__(location, manager, capacity, occupied_spots, available_spo
        self.valet_count = valet_count

    def display_info(self):
        print("Valet Count:", self.valet_count)

    def park_vehicle(self):
        if self.available_spots > 0 and self.valet_count > 0:
            self.available_spots -= 1
            self.occupied_spots += 1
            self.valet_count -= 1
            print("Vehicle parked by valet successfully.")
        else:
            print("Parking or valet capacity is full.")
```

Creating objects and calling functions

In [ ]:
```
parking_system = VehicleParkingManagementSystem("Ground Floor Parking", "Brun Samer
parking_system.display_info()
parking_system.park_vehicle()
parking_system.vacate_spot()
print("----------------------------------------------------------------")
valet_system = ValetParkingSystem("1st Floor Parking", "Jane Smith", 30, 5, 25, 3)
valet_system.display_info()
valet_system.park_vehicle()
valet_system.vacate_spot()
```

```
Parking Location: Ground Floor Parking
Manager: Brun Samer
Total Capacity: 50
Occupied Spots: 10
Available Spots: 40
Vehicle parked successfully.
Spot vacated successfully.
----------------------------------------------------------------
Valet Count: 3
Vehicle parked by valet successfully.
Spot vacated successfully.
```