

**Abbottabad university of science & technology**

***HOSPITAL MANAGEMENT SYSTEM:***

**Submitted by : Fatima Bibi**

**Submitted to : Sir Jamal Abdul Ahad**

**Roll No : 14646**

**Course : Data structure & Algorithm**

**Submission date : January 31, 2025**

***HOSPITAL MANAGEMENT SYSTEM:***

***CODE:***

import heapq # Import heapq for heap operations

# Global Variables

Patient\_ID = []

Patient\_Name = []

Patient\_Age = []

Patient\_Gender = []

Diagnosis = []

Room\_Assignment = []

Doctor\_Name = []

Treatment = []

Billing = {}

# Heaps for room numbers

general\_rooms = list(range(1, 51)) # 50 rooms

icu\_rooms = list(range(51, 71)) # 20 rooms

emergency\_rooms = list(range(71, 81)) # 10 rooms

# Convert to heaps

heapq.heapify(general\_rooms)

heapq.heapify(icu\_rooms)

heapq.heapify(emergency\_rooms)

# Priority Queue for Emergency Cases

patient\_queue = deque()

emergency\_queue = deque()

# Doctor Availability

doctor\_availability = {

'Dr. Imran': True,

'Dr. Fatima Shah': True,

'Dr. Abdul Qadeer': True,

'Dr. Rehan': True,

'Dr. Ahmad': True

}

# Illness to room and doctor mapping

illness\_mapping = {

'fever': ('general', 'Dr. Imran', 500),

'cold': ('general', 'Dr. Fatima Shah', 300),

'heart attack': ('icu', 'Dr. Abdul Qadeer', 5000),

'stroke': ('icu', 'Dr. Rehan', 4500),

'accident': ('emergency', 'Dr. Ahmad', 7000)

}

def suggest\_treatment(diagnosis):

"""Suggests a treatment based on the diagnosis."""

treatments = {

'fever': 'Paracetamol and rest',

'cold': 'Antihistamines and fluids',

'heart attack': 'Immediate surgery and medication',

'stroke': 'Clot-busting drugs and therapy',

'accident': 'Emergency surgery and wound care'

}

return treatments.get(diagnosis, "No treatment available")

def get\_patient\_id():

"""Get and validate the patient ID."""

while True:

pid = input("\tEnter patient ID (XXXX-XXXX): ").upper()

if len(pid) == 9 and pid not in Patient\_ID:

return pid

elif pid in Patient\_ID:

print("###### Patient ID Already Exists ######")

else:

print("###### Enter a Valid Patient ID ######")

def get\_patient\_info():

"""Get and validate patient information."""

name = input("\tEnter patient name: ")

age = input("\tEnter patient age: ")

gender = input("\tEnter patient gender (M/F): ").upper()

return name, age, gender

def get\_diagnosis():

"""Get the diagnosis."""

diagnosis = input("\tEnter diagnosis (fever/cold/heart attack/stroke/accident): ").lower()

while diagnosis not in illness\_mapping:

print("###### Invalid illness. Please choose from fever, cold, heart attack, stroke, or accident ######")

diagnosis = input("\tEnter diagnosis: ").lower()

return diagnosis

def assign\_room\_and\_doctor(diagnosis):

"""Assign room and doctor based on diagnosis."""

room\_type, doctor, cost = illness\_mapping[diagnosis]

room = None

if room\_type == "general" and general\_rooms:

room = heapq.heappop(general\_rooms)

elif room\_type == "icu" and icu\_rooms:

room = heapq.heappop(icu\_rooms)

elif room\_type == "emergency" and emergency\_rooms:

room = heapq.heappop(emergency\_rooms)

if room:

doctor\_availability[doctor] = False

return room, doctor, cost

else:

print("###### No rooms available for this diagnosis. Try another diagnosis. ######")

return None, None, None

def add\_patient():

"""Add a new patient entry."""

patient\_id = get\_patient\_id()

name, age, gender = get\_patient\_info()

diagnosis = get\_diagnosis()

room, doctor\_name, cost = assign\_room\_and\_doctor(diagnosis)

if room:

treatment = suggest\_treatment(diagnosis)

# Store patient information

Patient\_ID.append(patient\_id)

Patient\_Name.append(name)

Patient\_Age.append(age)

Patient\_Gender.append(gender)

Diagnosis.append(diagnosis)

Room\_Assignment.append(room)

Doctor\_Name.append(doctor\_name)

Treatment.append(treatment)

Billing[patient\_id] = cost

# Add patient to appropriate queue

if diagnosis in ["heart attack", "stroke", "accident"]:

emergency\_queue.append(patient\_id)

else:

patient\_queue.append(patient\_id)

print(f"\n###### Patient assigned to Room {room} under {doctor\_name}. Treatment: {treatment}. Estimated Cost: Rs.{cost}. Entry Saved ######")

else:

print("###### Unable to assign a room. Patient not added. ######")

def process\_patient():

"""Process the next patient in the queue."""

if emergency\_queue:

pid = emergency\_queue.popleft()

elif patient\_queue:

pid = patient\_queue.popleft()

else:

print("No patients in queue.")

return

print(f"Processing patient {pid}. Preparing for doctor's consultation and treatment.")

def discharge\_patient():

"""Remove a patient and free up resources."""

pid = input("Enter Patient ID to discharge: ").upper()

if pid in Patient\_ID:

index = Patient\_ID.index(pid)

room = Room\_Assignment[index]

doctor = Doctor\_Name[index]

# Free the room

if room in range(1, 51):

heapq.heappush(general\_rooms, room)

elif room in range(51, 71):

heapq.heappush(icu\_rooms, room)

else:

heapq.heappush(emergency\_rooms, room)

# Free the doctor

doctor\_availability[doctor] = True

# Process payment

payment = Billing[pid]

print(f"###### Payment Processed. Total Amount: Rs.{payment} ######")

# Remove patient

Patient\_ID.pop(index)

Patient\_Name.pop(index)

Patient\_Age.pop(index)

Patient\_Gender.pop(index)

Diagnosis.pop(index)

Room\_Assignment.pop(index)

Doctor\_Name.pop(index)

Treatment.pop(index)

Billing.pop(pid, None)

print("\n###### Patient Discharged ######")

else:

print("###### Patient Not Found ######")

def search\_patient():

"""Search for a patient by their ID."""

pid = input("Enter Patient ID to search: ").upper()

if pid in Patient\_ID:

index = Patient\_ID.index(pid)

print(f"Patient found: {Patient\_Name[index]} - {Diagnosis[index]}")

else:

print("###### Patient Not Found ######")

import random

def check\_doctor\_availability():

"""Check if a doctor is available."""

doctor\_name = input("Enter doctor's name: ").strip()

# Randomly decide doctor availability (True or False)

is\_available = random.choice([True, False])

if is\_available:

print(f"{doctor\_name} is available for consultation.")

else:

print(f"{doctor\_name} is not available at the moment.")

def view\_patients():

"""View all patient records."""

print("\nPatient ID | Name | Age | Gender | Diagnosis | Room | Doctor | Treatment | Cost")

for i in range(len(Patient\_ID)):

print(f"{Patient\_ID[i]} | {Patient\_Name[i]} | {Patient\_Age[i]} | {Patient\_Gender[i]} | {Diagnosis[i]} | {Room\_Assignment[i]} | {Doctor\_Name[i]} | {Treatment[i]} | ${Billing.get(Patient\_ID[i], 'N/A')}")

def main():

"""Main menu loop."""

while True:

print("\n1. Add Patient")

print("2. Process Next Patient")

print("3. Discharge Patient")

print("4. View Patients")

print("5. Search Patient")

print("6. Check Doctor Availability")

print("7. Exit")

choice = input("Choose an option: ")

if choice == "1":

add\_patient()

elif choice == "2":

process\_patient()

elif choice == "3":

discharge\_patient()

elif choice == "4":

view\_patients()

elif choice == "5":

search\_patient()

elif choice == "6":

check\_doctor\_availability()

elif choice == "7":

print("Exiting... Goodbye!")

break

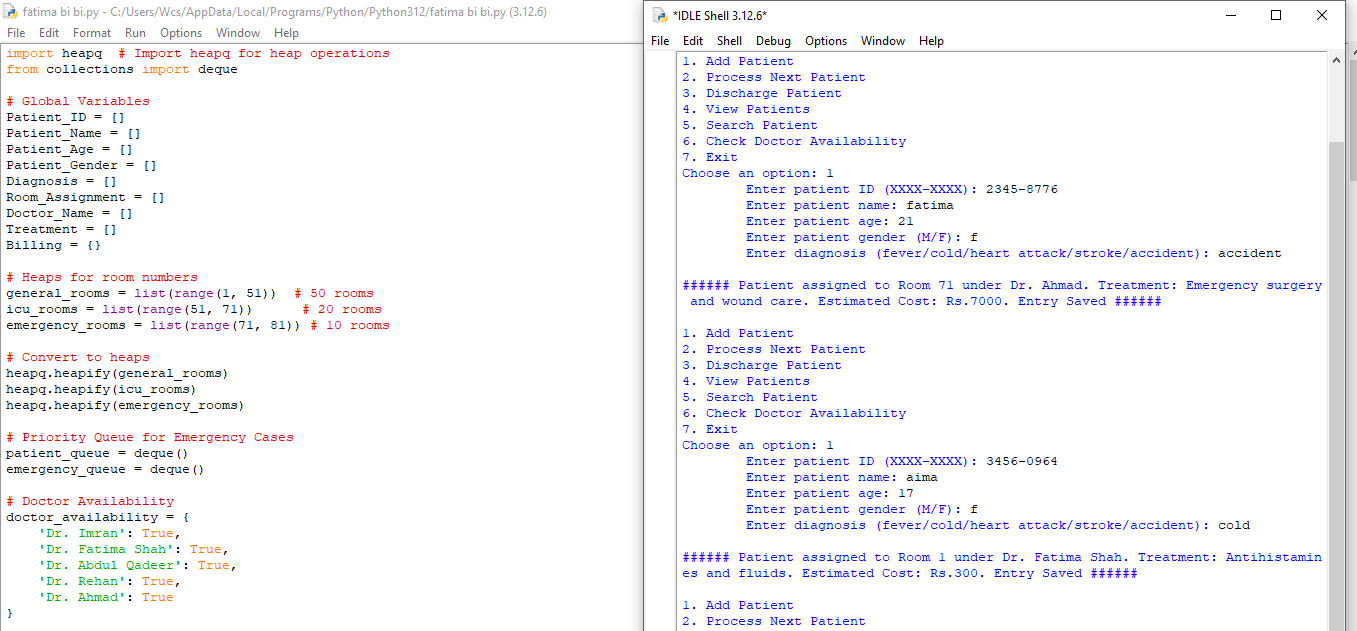
else:

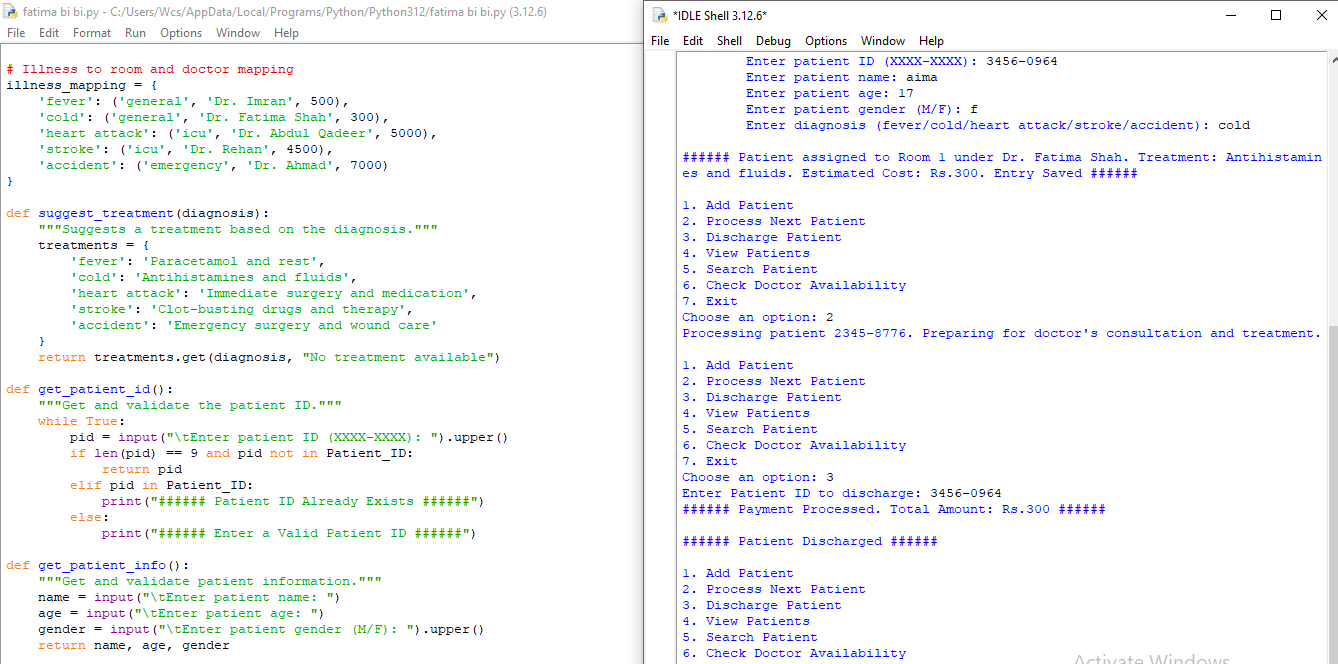
print("Invalid Option!")

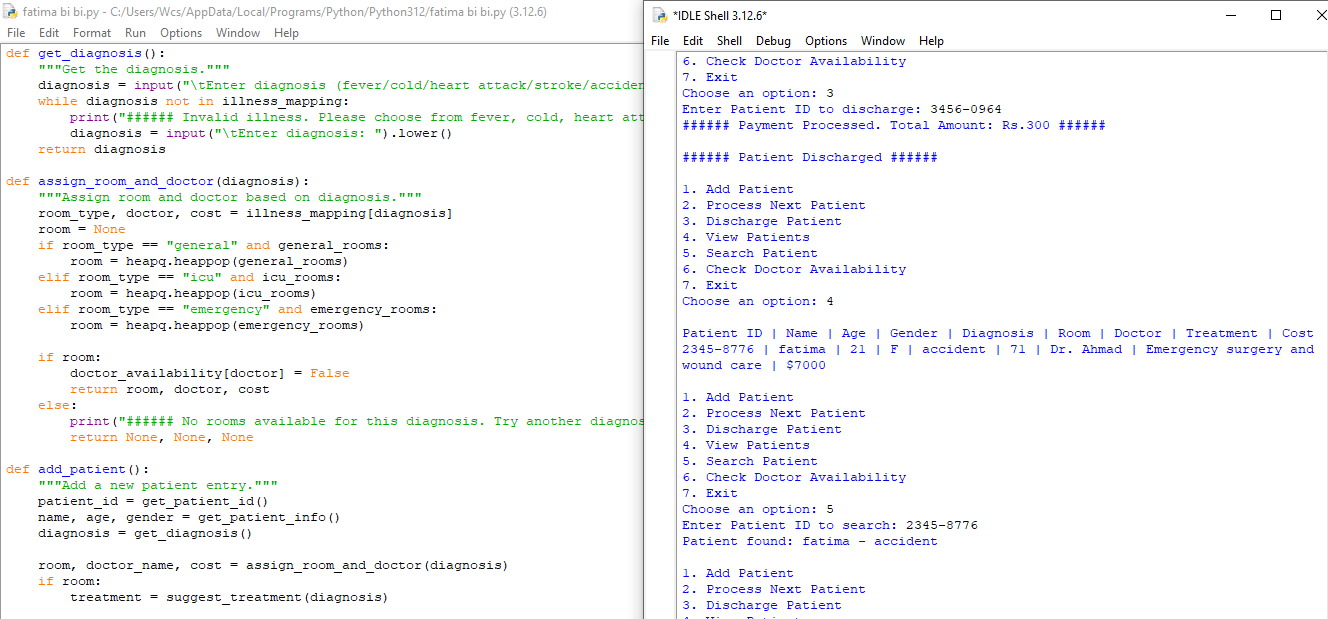
if \_\_name\_\_ == "\_\_main\_\_":

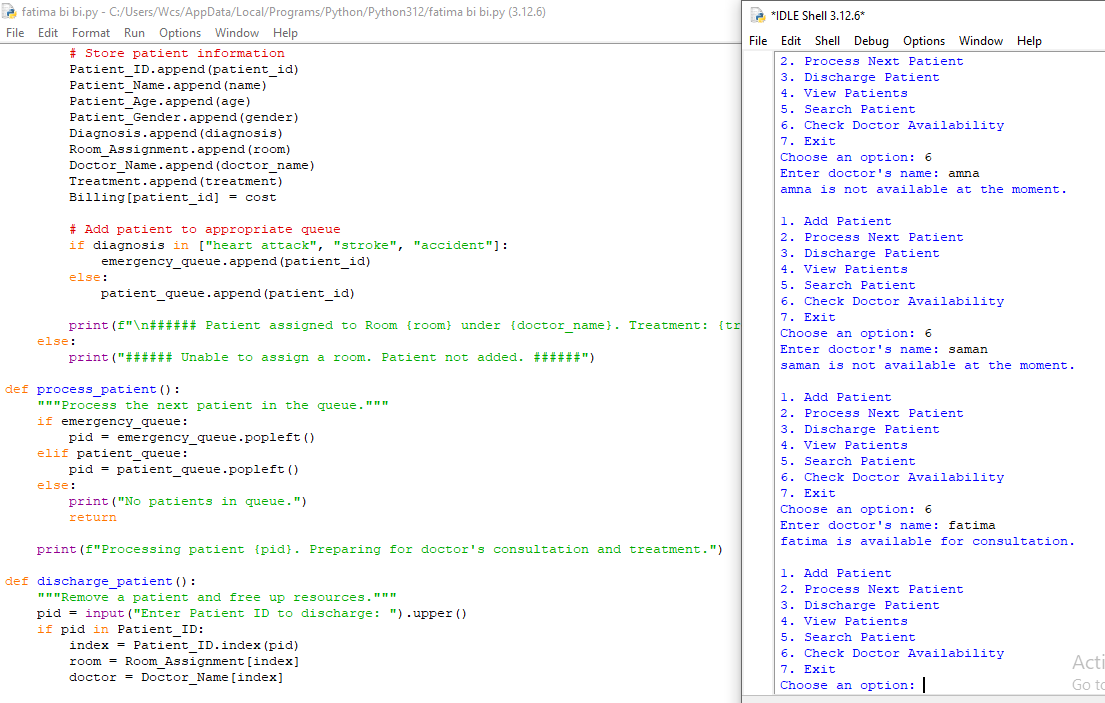
main()

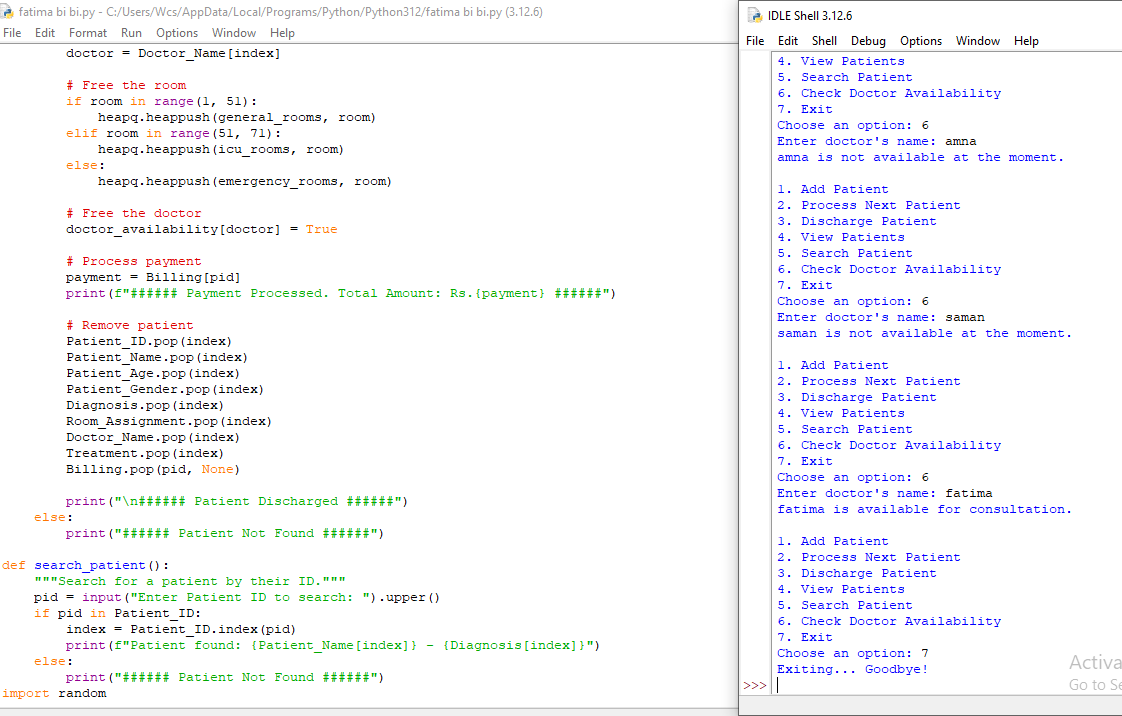
***CODE AND OUTPUT SCREEN SHOTS:***

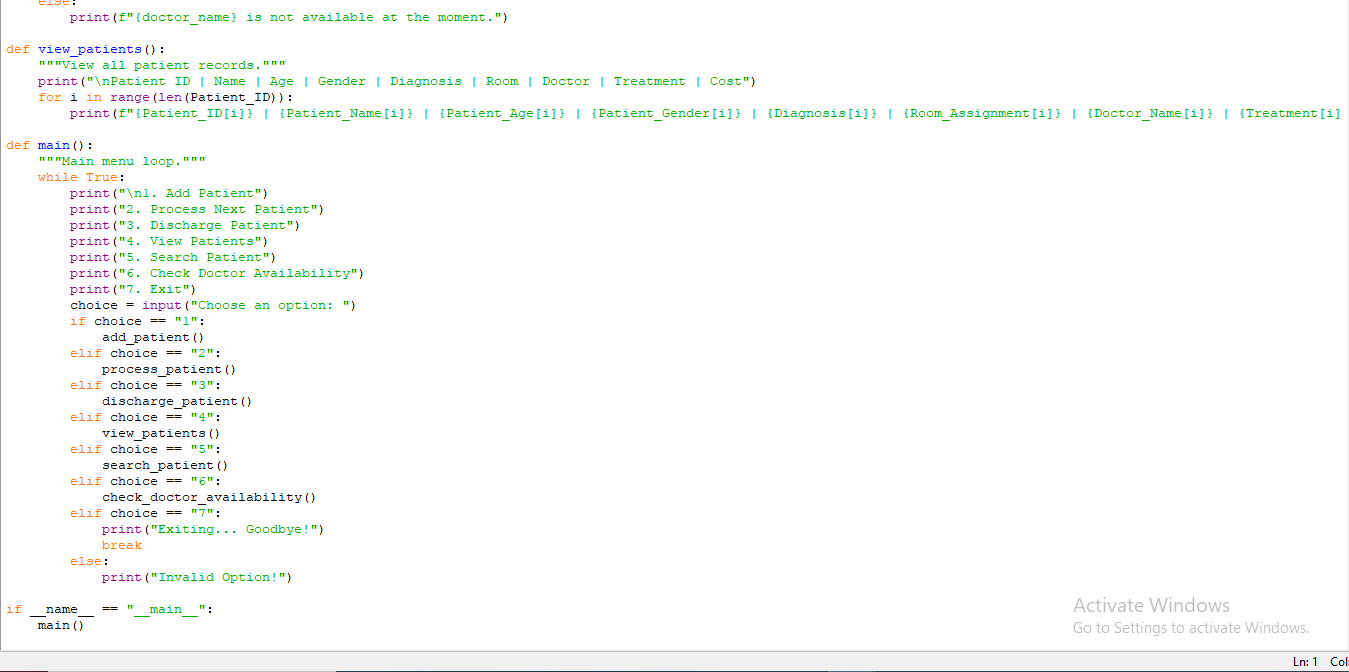












***OUTPUT :***

