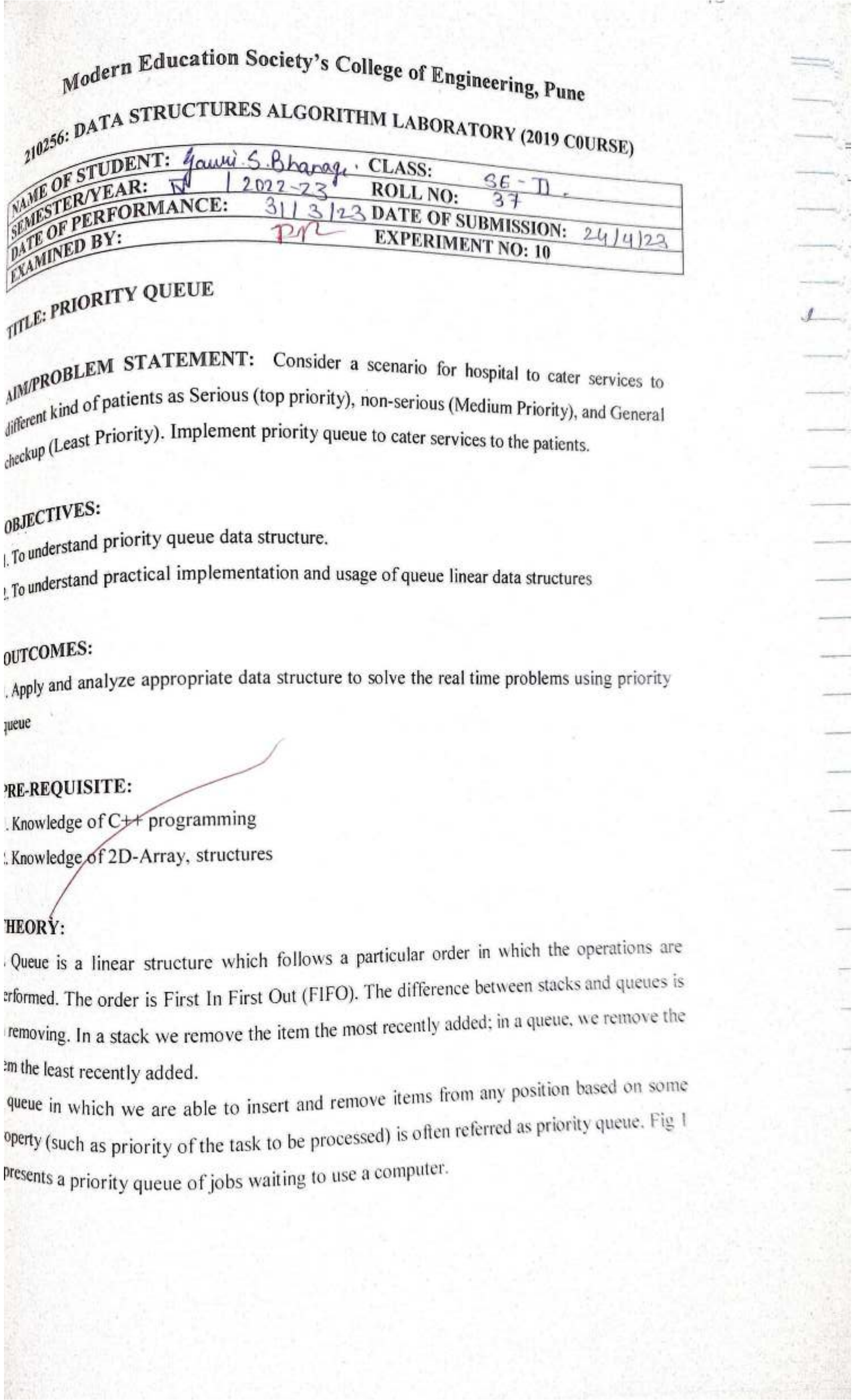


Modern Education Society's College of Engineering, Pune

210256: DATA STRUCTURES ALGORITHM LABORATORY (2019 C0URSE)

NAME OF STUDENT: Gauri S. Bhanage.	CLASS: SE - II.
SEMESTER/YEAR: IV | 2022-23	ROLL NO: 37
DATE OF PERFORMANCE: 31 | 3 | 23	DATE OF SUBMISSION: 24 | 4 | 23
EXAMINED BY: PR	EXPERIMENT NO: 10

TITLE: PRIORITY QUEUE

AIM/PROBLEM STATEMENT: Consider a scenario for hospital to cater services to different kind of patients as Serious (top priority), non-serious (Medium Priority), and General checkup (Least Priority). Implement priority queue to cater services to the patients.

OBJECTIVES:

1. To understand priority queue data structure.
2. To understand practical implementation and usage of queue linear data structures

OUTCOMES:

. Apply and analyze appropriate data structure to solve the real time problems using priority queue

'RE-REQUISITE:

. Knowledge of C++ programming
. Knowledge of 2D-Array, structures

HEORY:

. Queue is a linear structure which follows a particular order in which the operations are erformed. The order is First In First Out (FIFO). The difference between stacks and queues is removing. In a stack we remove the item the most recently added; in a queue, we remove the em the least recently added.

queue in which we are able to insert and remove items from any position based on some operty (such as priority of the task to be processed) is often referred as priority queue. Fig 1 presents a priority queue of jobs waiting to use a computer.

Theory.

A priority queue is a queue that arranges elements based on their priority values. Elements with higher priority values are typically retrived before elements with lower priority values.

In priority queue, each element has a priority value associated with it.

when you add an element to the queue, it is inserted in a position based on its priority value

There are several ways to implement a priority queue, including using an array, linked list, heap or binary search tree.

- Properties of Priority Queue

•) Every item has apriority associated with it

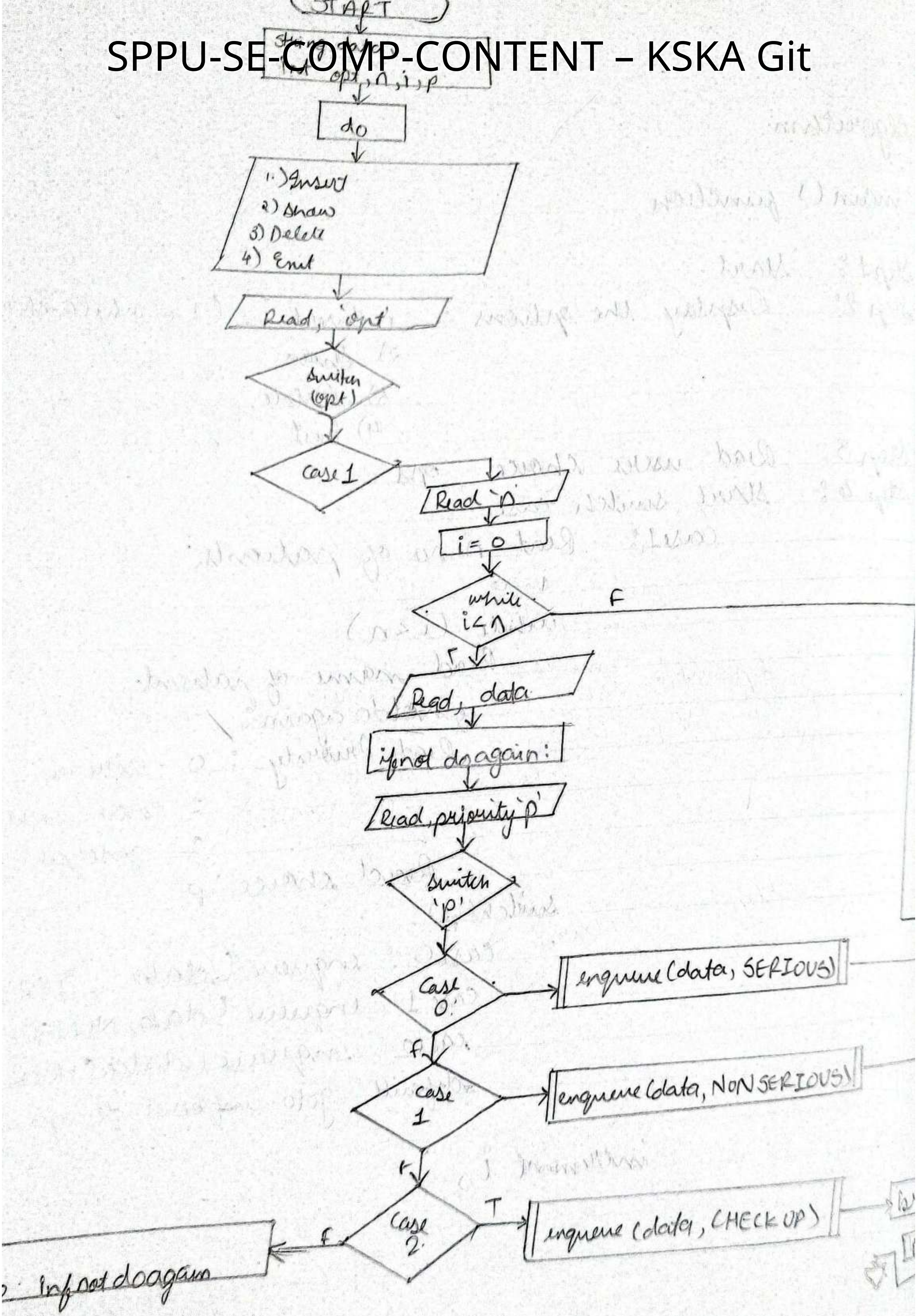
•) An item / element with high prioprity is dequeued before an element with low priority

•) If two elements have same priority, they are served according to their order in sequence.

Algorithm

main() function

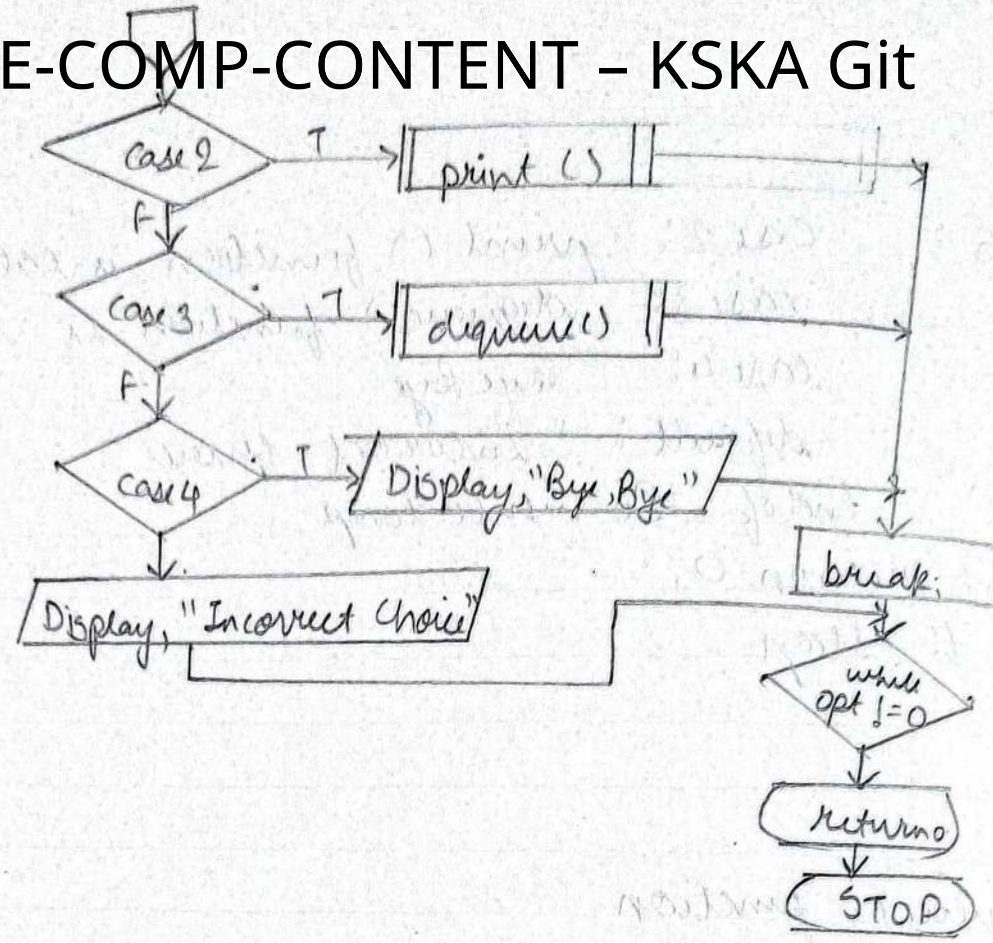
```
Step1:- Start
Step 2:  Display the options :-  1) Insert   (Do-while started)
                                 2) Show
                                 3) Delete
                                 4) Exit
Step 3:  Read users choice `opt'
Step 4:- Start switch case.
         case1:- Read `n' no. of patients
                 i=0
                 while (i<n)
                     Read name of patient
                     ifnotdo again:
                     Read Priority : 0 - serious
                                     1 - non serious
                                     2 - general checkup
                         Read choice 'p'
                 switch(p)
                     case0 : enqueue (data , SERIOUS)
                     case 1 : enqueue (data, NONSERIOUS)
                     case2 : enqueue (data, CHECK UP),
                     default : goto ifnot doagain

         increment i;
```


START
string data
int opt, n, i, p
do
1.) Insert
2) Show
3) Delete
4) Exit
Read, 'opt'
Switch (opt)
case 1
Read 'n'
i = 0
while i<n
F
T
Read, data
if not do again:
Read, priority 'p'
Switch 'p'
Case 0
enqueue(data, SERIOUS)
F
case 1
enqueue(data, NONSERIOUS)
F
Case 2
T
enqueue(data, CHECKUP)
F
: Inf not doagain

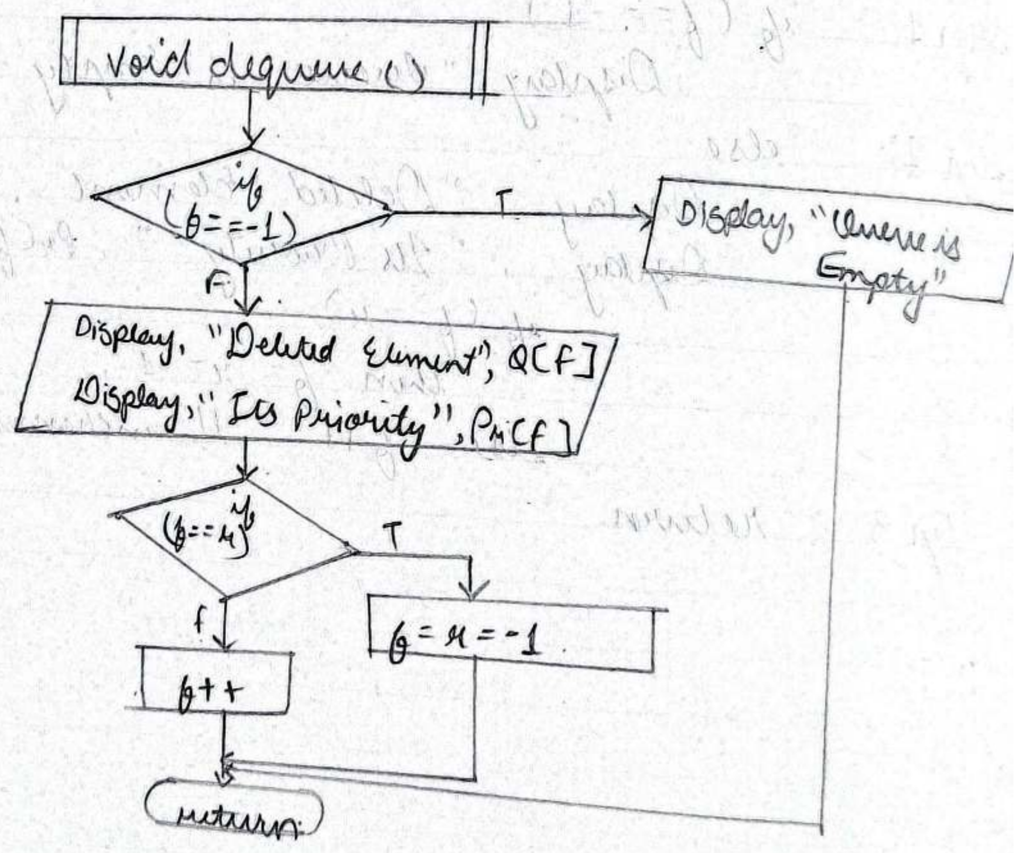
```
p5:-    case 2:  print () function is called.
        case 3:  dequeue() function is called.
        case 4:  "Bye Bye"
        default : "Incorrect choice"
      End of  Do-while loop
n6:- return 0;
 7:- stop
```

queue() function

```
n1:   if (f==-1)
           Display "Queue is empty"
 2:-   else.
         Display , "Deleted Element", Q[f] ;
         Display , "Its Priority =", Pr[f].
              if (f==r)
                   then f= r=-1 ,
              else , f++    // increment f
p3 :- return
```


case 2
T
F
print ()
case 3
T
F
dequeue()
case 4
T
Display, "Bye, Bye"
Display, "Incorrect Choice"
break;
while opt != 0
return 0
STOP

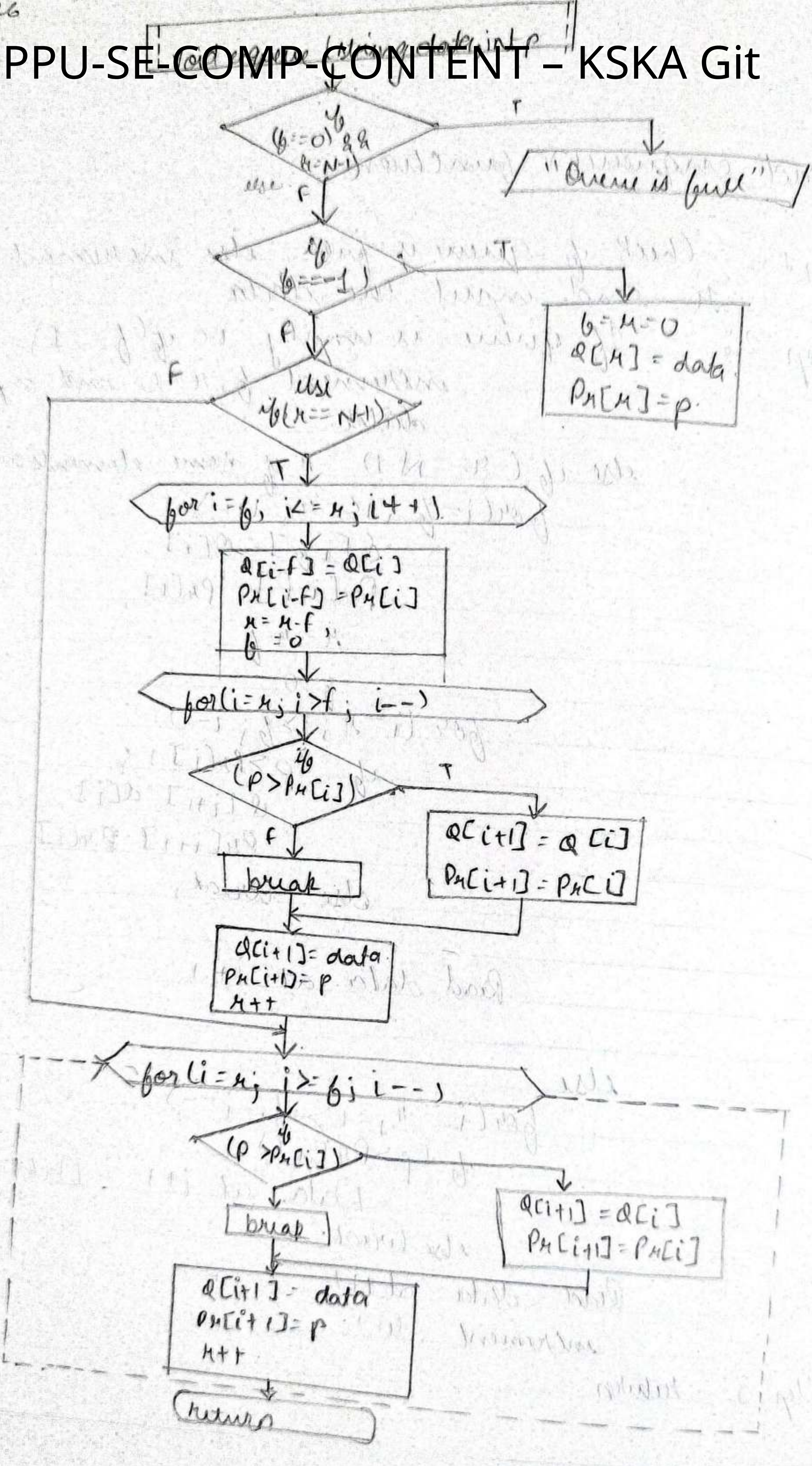
2) dequeue()


void dequeue ()
if (f==-1)
T
Display, "Queue is Empty"
F
Display, "Deleted Element", Q[f]
Display, "Its Priority", Pr[f]
if (f==r)
T
f
f = r = -1
f++
return

void enqueue() function

```
Step 1:-  Check if queue is full, else increment f and
          r and input the data.
Step 2:-  If queue is empty i.e. if(f==-1)
                  increment f, r=0 and input
                  data.
          else if ( r==N-1)   // if some elements are present
              for(i=f ; i<=r ; i++)
                      Q[i-f] = Q[i]
                      Pr[i-f] = Pr[i];
                       r = r-f ;
                       f=0
              for (i=r ; i>f ; i--)
                   if  (p>Pr[i]) {
                          Q[i+1]=Q[i];
                          Pr[i+1]=Pr[i]
                   else break;

              Read data at i+1

          else
              for(i=r; i>=f; i--)
                   if (p>Pr[i])
                        Data at i+1 = Data at i
                   else break
          Read data at i+1
          increment r++
Step 3:- return.
```

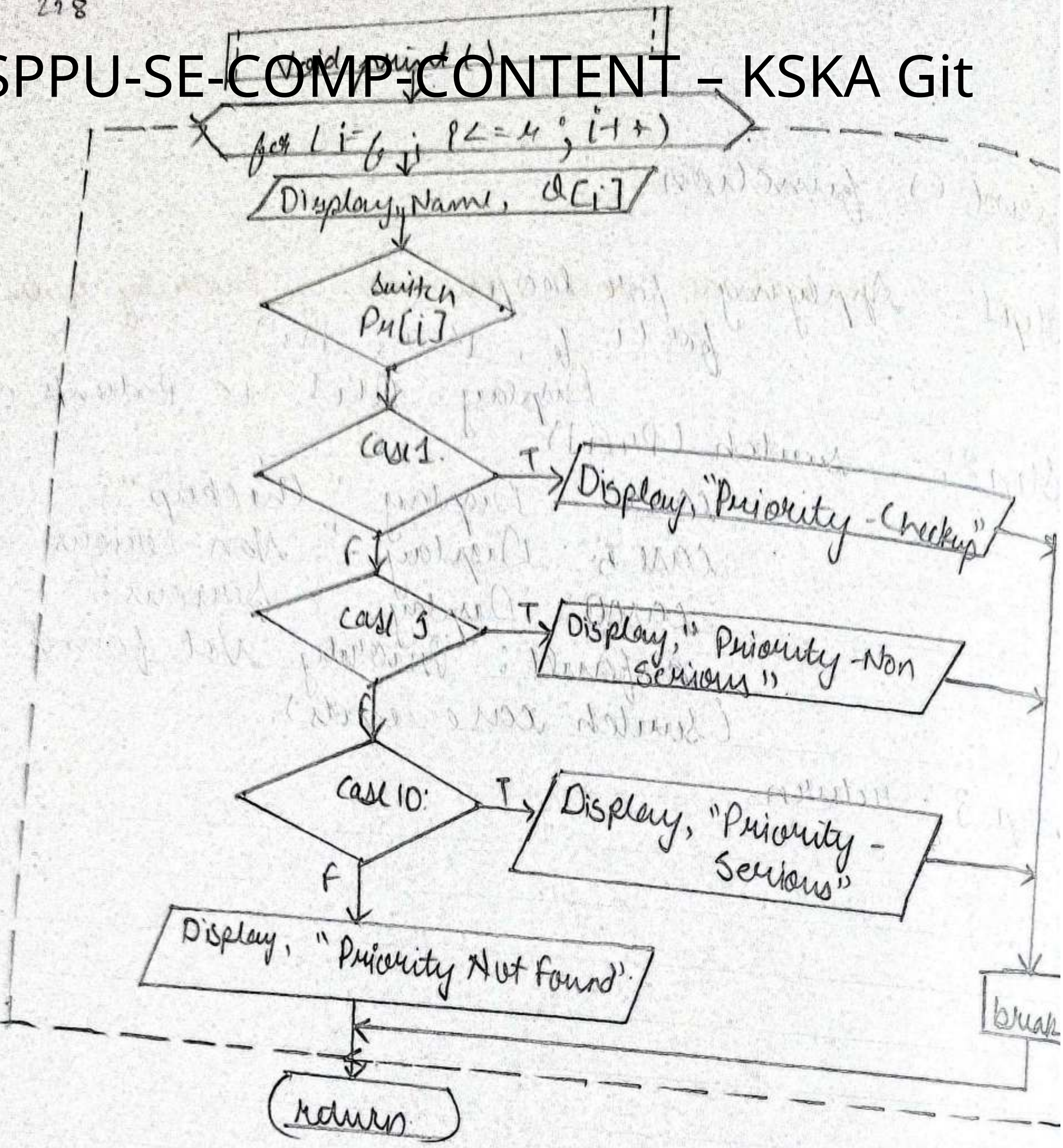
3] 226


void enqueue (int data, int p)
if (f==0) && (r==N-1)
T
"Queue is full"
else
F
if (f==-1)
T
f=r=0
Q[r] = data
Pr[r] = p
F
else if(r==N-1)
F
T
for i=f; i<=r; i++)
Q[i-f] = Q[i]
Pr[i-f] =Pr[i]
r= r-f
f =0 ;
for(i=r; i>f ; i--)
if (p>Pr[i])
T
Q[i+1] = Q [i]
Pr[i+1] = Pr[i]
f
break
Q[i+1]= data
Pr[i+1]= p
r++
for(i=r; i>=f; i--)
if (p >Pr[i])
break
Q[i+1] =Q[i]
Pr[i+1] = Pr[i]
Q[i+1] = data
Pr[i+1]= p
r++
return

4) print () function

```
Step1:- Applying for loop
        for (i=f; i<=r, i++)
            Display Q[i] ie Patients name
Step2:- switch (Pr[i])
        case1:- Display "Checkup";
        case 2: Display " Non-Serious"
        case 3: Display " Serious"
        default: "Priority Not found"
        (Switch case ends)
Step 3:- return.
```

4]


void print()
for (i=0; i<=4; i++)
Display, Name, Q[i]
switch
Pn[i]
case1.
T
Display, "Priority-Checkup"
F
case 5
T
Display," Priority-Non Serious"
F
case 10.
T
Display, "Priority - Serious"
F
Display, "Priority Not found".
break
return

Questions

1) Define Ascending and Descending Priority queue.

1) Ascending Order Priority Queue

In this the element with a lower priority value is given a higher priority in the priority list. For example, if we have the following elements in a priority queue arranged in ascending order like 4,6,8,9,10. Here, 4 is the smallest number, therefore it will get the highest priority in a priority queue and so when we dequeue from this type of priority queue, 4 will remove from the queue and dequeue returns 4.

2) Descending Order Priority Queue.

The root node is a maximum element in a max heap, as you may know. It will also remove the element with the highest priority first. As a result, the root node is removed from the queue. This deletion leaves an empty space, which will be filled with fresh insertions in future.

2) Describe various applications of Priority Queue

ns ·) CPU scheduling
·) Graph algorithms
·) Stack implementation

- All queue applications where priority queue is involved
- Data compression in Huffman code.
- Event-driven simulation such as customers waiting in aqueue

3.) Write short note on ADT of Priority queue.

The Priority Queue ADT is designed for systems that maintaining a collection of prioritized elements, where elements are removed from the collection in order of their priority

The priority queue ADT includes four operations:

- add()
- isEmpty()
- remove ()
- peek.

Conclusion

Hence successfully implmented and understood the data structures of priority queue.