Bharatiya Vidya Bhavan's
Sardar Patel Institute of Technology
(Autonomous Institute Affiliated to University of Mumbai)

Name: Jhaveri Varun Nimitt

UID: 2023800042

Batch: CSE A Batch C

Experiment No.:2

Aim: Queue application

Problem:

Find the first circular tour that visits all petrol pumps

Suppose there is a circle. There are n petrol pumps on that circle. You are given two sets of data.

i. The amount of petrol that every petrol pump has.

ii. Distance from that petrol pump to the next petrol pump.

Questions:

a. Design and implement the given scenario using circular queue

b. Find the first point from where a truck will be able to complete the circle (The truck will stop at each petrol pump and it has infinite capacity).

C. output all possible successful tours

Assume for 1 litre petrol, the truck can go 1 unit of distance.

For example, let there be 4 petrol pumps with amount of petrol and distance to next petrol pump value pairs as {4, 6}, {6, 5}, {7, 3} and {4, 5}. The first point from where truck can make a circular tour is 2nd petrol pump. Output should be starting petrol pump = 1(index of 2nd petrol pump)and complete tour is 1->2->3->0->1

Program:

```c
#include <stdio.h>
#define MAX 100

typedef struct {
   int petrol;
   int distance;
} PetrolPump;

typedef struct {
   PetrolPump pumps[MAX];
   int front;
   int rear;
   int size;
} CircularQueue;

void enqueue(CircularQueue* q, PetrolPump item, int debug) {
   if (q->size == MAX) {
       if (debug) printf("DEBUG: Queue is full.\n");
       return;
   }
   if (q->front == -1) {
       q->front = 0;
   }
   q->rear = (q->rear + 1) % MAX;
   q->pumps[q->rear] = item;
   q->size++;
   if (debug) printf("DEBUG: Enqueued: petrol = %d, distance = %d at position %d\n", item.petrol,
item.distance, q->rear);
}

PetrolPump dequeue(CircularQueue* q, int debug) {
   PetrolPump item = {0, 0};
   if (q->size == 0) {
       if (debug) printf("DEBUG: Queue is empty.\n");
       return item;
   }
   item = q->pumps[q->front];
   q->front = (q->front + 1) % MAX;
   q->size--;
   if (debug) printf("DEBUG: Dequeued: petrol = %d, distance = %d from position %d\n",
item.petrol, item.distance, q->front);
   return item;
```

```
}

int isTourPossible(CircularQueue* q, int n, int debug) {
    int fuel = 0;
    int count = 0;
    int start = q->front;

    if (debug) printf("DEBUG: Checking tour possibility starting at index %d\n", start);

    while (count < n) {
        PetrolPump current = dequeue(q, debug);
        fuel += current.petrol;

        if (debug) printf("DEBUG: Current fuel after adding pump = %d\n", fuel);

        if (fuel < current.distance) {
            if (debug) printf("DEBUG: Not enough fuel to cover distance from pump (petrol = %d,
distance = %d). Tour is impossible.\n", current.petrol, current.distance);
            return 0;
        }

        fuel -= current.distance;
        if (debug) printf("DEBUG: Fuel after covering distance = %d\n", fuel);

        enqueue(q, current, debug);
        count++;
    }
    if (debug) printf("DEBUG: Tour is possible starting from index %d\n", start);
    return 1;
}

int findFirstTour(PetrolPump pumps[], int n, int debug) {
    CircularQueue q = {.front = -1, .rear = -1, .size = 0};

    for (int i = 0; i < n; i++) {
        enqueue(&q, pumps[i], debug);
    }

    for (int i = 0; i < n; i++) {
        if (isTourPossible(&q, n, debug)) {
            return i;
        }

        enqueue(&q, dequeue(&q, debug), debug);
    }
    return -1;
}
```

```c
void displayTour(int start, int n) {
    printf("Tour: ");
    for (int i = 0; i < n; i++) {
        printf("%d->", (start + i) % n);
    }
    printf("%d\n", start);
}

void findAllPossibleTours(PetrolPump pumps[], int n, int debug) {
    CircularQueue q = {.front = -1, .rear = -1, .size = 0};

    for (int i = 0; i < n; i++) {
        enqueue(&q, pumps[i], debug);
    }

    for (int i = 0; i < n; i++) {
        if (isTourPossible(&q, n, debug)) {
            printf("Starting pump = %d\n", i);
            displayTour(i, n);
        }

        enqueue(&q, dequeue(&q, debug), debug);
    }
}

int main() {
    PetrolPump pumps[] = {{4, 6}, {6, 5}, {7, 3}, {4, 5}};
    int n = sizeof(pumps) / sizeof(pumps[0]);
    int debug = 1;

    int firstTour = findFirstTour(pumps, n, debug);
    if (firstTour != -1) {
        printf("Start for first proper tour = %d\n", firstTour);
        displayTour(firstTour, n);
    } else {
        printf("No possible tour found.\n");
    }

    printf("All possible tours:\n");
    findAllPossibleTours(pumps, n, debug);

    return 0;
}
```

Task1:Handwritten assignment submission:

Show the enqueue operation of the given problem step-wise

2021800042

Circular Queue Illustration

→ Queue Properties
- MAX = 100 = size of queue (we are only using 4)
- front points to index of first element
- rear points to index of last element
- size = no of elements

→ Initial state
[ ]
front = -1
rear = -1
size = 0

→ Enqueue Print 1
- Queue is empty set front to 0
- Rear = (rear+1) % max = (-1+1) % 100 = 0

∴ Queue
[4,6]

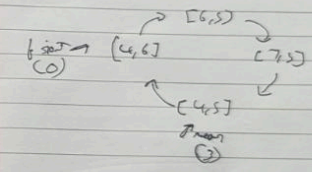
- Size 1

→ Enqueue Enqueue Print 2
- Rear = 1
- Front = 0
- size = 2
- Queue
[4,6] → [6,5]

→ Enqueue print 3
- Rear = 2
- Front = 0
- Size = 3
- Queue
[4,6] → [6,5] → [7,5]

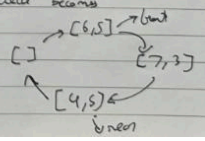
→ Enqueue print 4
- Rear = 3
- Front = 0
- Size = 4

∴ Queue full



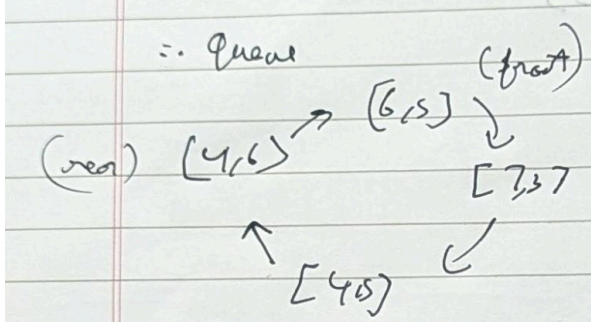
→ ~~Enqueue~~ Dequeue something
- After dequeue front = (front+1) % max ∴ 1
- rear remains at 3
- size --;

∴ Queue becomes

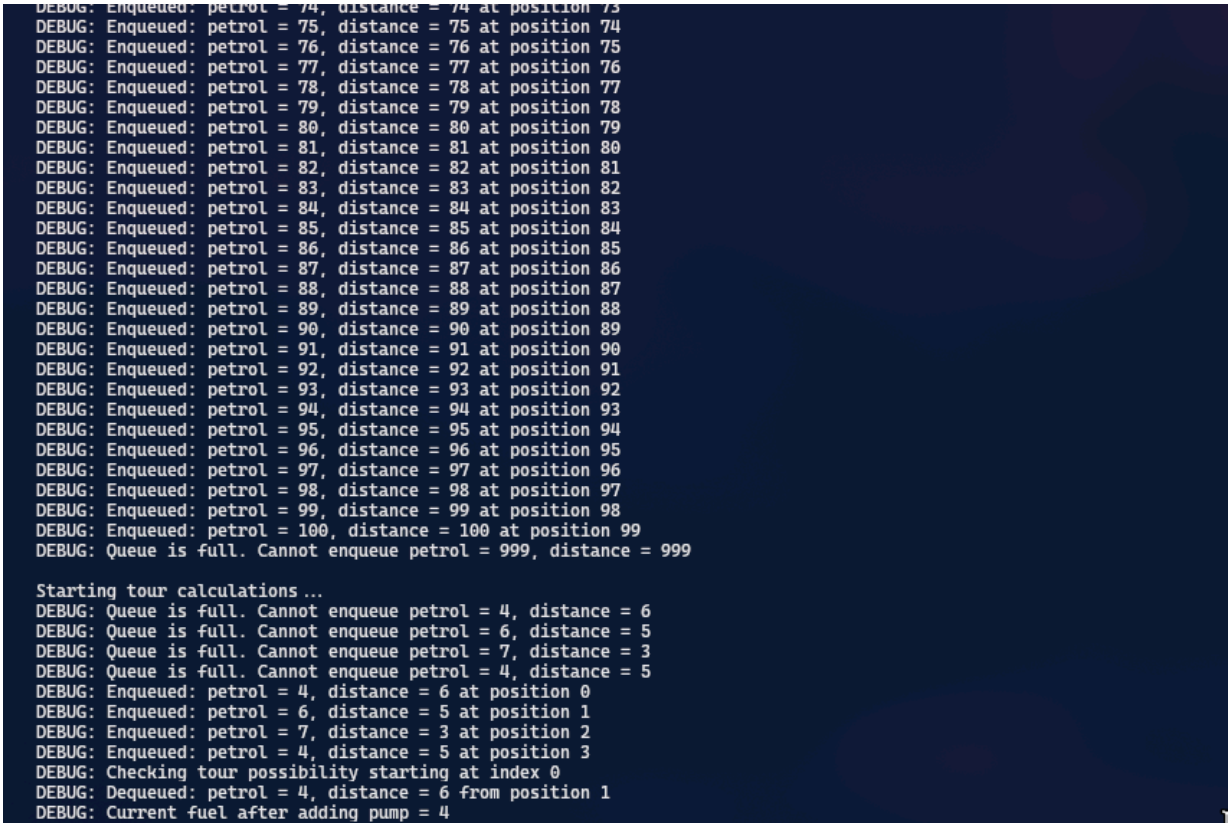


→ Adding dequeued element back
- rear = (rear+1) % max = 0 since it wraps around
→ in our case this is 4
- 4,6 is added at index 0
- size ++

∴ Queue



Output no 1 (forcefully enqueueing if queue is full):**(code)**



```
DEBUG: Enqueued: petrol = 74, distance = 74 at position 73
DEBUG: Enqueued: petrol = 75, distance = 75 at position 74
DEBUG: Enqueued: petrol = 76, distance = 76 at position 75
DEBUG: Enqueued: petrol = 77, distance = 77 at position 76
DEBUG: Enqueued: petrol = 78, distance = 78 at position 77
DEBUG: Enqueued: petrol = 79, distance = 79 at position 78
DEBUG: Enqueued: petrol = 80, distance = 80 at position 79
DEBUG: Enqueued: petrol = 81, distance = 81 at position 80
DEBUG: Enqueued: petrol = 82, distance = 82 at position 81
DEBUG: Enqueued: petrol = 83, distance = 83 at position 82
DEBUG: Enqueued: petrol = 84, distance = 84 at position 83
DEBUG: Enqueued: petrol = 85, distance = 85 at position 84
DEBUG: Enqueued: petrol = 86, distance = 86 at position 85
DEBUG: Enqueued: petrol = 87, distance = 87 at position 86
DEBUG: Enqueued: petrol = 88, distance = 88 at position 87
DEBUG: Enqueued: petrol = 89, distance = 89 at position 88
DEBUG: Enqueued: petrol = 90, distance = 90 at position 89
DEBUG: Enqueued: petrol = 91, distance = 91 at position 90
DEBUG: Enqueued: petrol = 92, distance = 92 at position 91
DEBUG: Enqueued: petrol = 93, distance = 93 at position 92
DEBUG: Enqueued: petrol = 94, distance = 94 at position 93
DEBUG: Enqueued: petrol = 95, distance = 95 at position 94
DEBUG: Enqueued: petrol = 96, distance = 96 at position 95
DEBUG: Enqueued: petrol = 97, distance = 97 at position 96
DEBUG: Enqueued: petrol = 98, distance = 98 at position 97
DEBUG: Enqueued: petrol = 99, distance = 99 at position 98
DEBUG: Enqueued: petrol = 100, distance = 100 at position 99
DEBUG: Queue is full. Cannot enqueue petrol = 999, distance = 999

Starting tour calculations...
DEBUG: Queue is full. Cannot enqueue petrol = 4, distance = 6
DEBUG: Queue is full. Cannot enqueue petrol = 6, distance = 5
DEBUG: Queue is full. Cannot enqueue petrol = 7, distance = 3
DEBUG: Queue is full. Cannot enqueue petrol = 4, distance = 5
DEBUG: Enqueued: petrol = 4, distance = 6 at position 0
DEBUG: Enqueued: petrol = 6, distance = 5 at position 1
DEBUG: Enqueued: petrol = 7, distance = 3 at position 2
DEBUG: Enqueued: petrol = 4, distance = 5 at position 3
DEBUG: Checking tour possibility starting at index 0
DEBUG: Dequeued: petrol = 4, distance = 6 from position 1
DEBUG: Current fuel after adding pump = 4
```

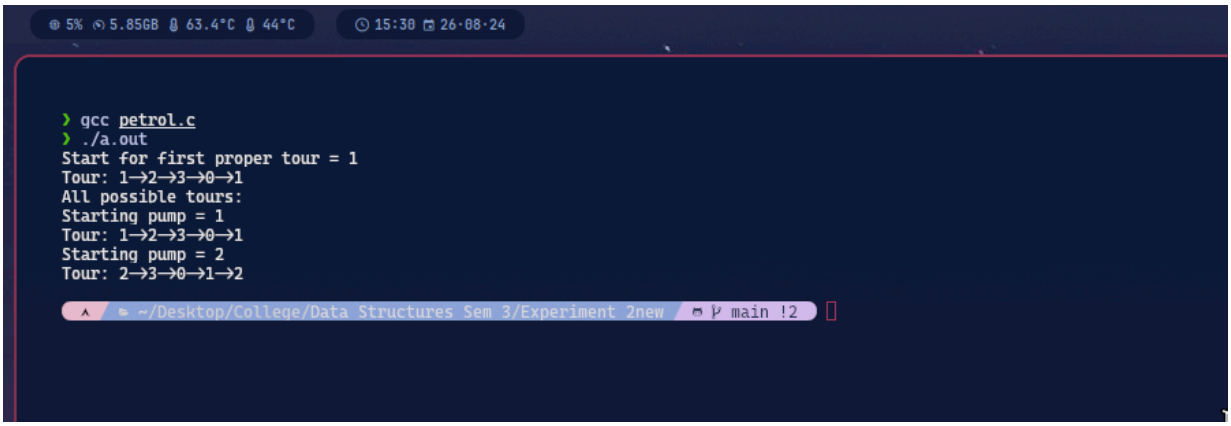
Output no 2: normal output with debug on (showcasing enqueue and dequeue operations)

```
 9% 5.81GB 53.4°C 44°C   15:29 26-08-24

❯ ./a.out
DEBUG: Enqueued: petrol = 4, distance = 6 at position 0
DEBUG: Enqueued: petrol = 6, distance = 5 at position 1
DEBUG: Enqueued: petrol = 7, distance = 3 at position 2
DEBUG: Enqueued: petrol = 4, distance = 5 at position 3
DEBUG: Checking tour possibility starting at index 0
DEBUG: Dequeued: petrol = 4, distance = 6 from position 1
DEBUG: Current fuel after adding pump = 4
DEBUG: Not enough fuel to cover distance from pump (petrol = 4, distance = 6). Tour is impossible.
DEBUG: Dequeued: petrol = 6, distance = 5 from position 2
DEBUG: Enqueued: petrol = 6, distance = 5 at position 4
DEBUG: Checking tour possibility starting at index 2
DEBUG: Dequeued: petrol = 7, distance = 3 from position 3
DEBUG: Current fuel after adding pump = 7
DEBUG: Fuel after covering distance = 4
DEBUG: Enqueued: petrol = 7, distance = 3 at position 5
DEBUG: Dequeued: petrol = 4, distance = 5 from position 4
DEBUG: Current fuel after adding pump = 8
DEBUG: Fuel after covering distance = 3
DEBUG: Enqueued: petrol = 4, distance = 5 at position 6
DEBUG: Dequeued: petrol = 6, distance = 5 from position 5
DEBUG: Current fuel after adding pump = 9
DEBUG: Fuel after covering distance = 4
DEBUG: Enqueued: petrol = 6, distance = 5 at position 7
DEBUG: Dequeued: petrol = 7, distance = 3 from position 6
DEBUG: Current fuel after adding pump = 11
DEBUG: Fuel after covering distance = 8
DEBUG: Enqueued: petrol = 7, distance = 3 at position 8
DEBUG: Tour is possible starting from index 2
Start for first proper tour = 1
Tour: 1→2→3→0→1
All possible tours:
DEBUG: Enqueued: petrol = 4, distance = 6 at position 0
DEBUG: Enqueued: petrol = 6, distance = 5 at position 1
DEBUG: Enqueued: petrol = 7, distance = 3 at position 2
DEBUG: Enqueued: petrol = 4, distance = 5 at position 3
DEBUG: Checking tour possibility starting at index 0
DEBUG: Dequeued: petrol = 4, distance = 6 from position 1
DEBUG: Current fuel after adding pump = 4
DEBUG: Not enough fuel to cover distance from pump (petrol = 4, distance = 6). Tour is impossible.
DEBUG: Dequeued: petrol = 6, distance = 5 from position 2
DEBUG: Enqueued: petrol = 6, distance = 5 at position 4
DEBUG: Checking tour possibility starting at index 2
DEBUG: Dequeued: petrol = 7, distance = 3 from position 3
DEBUG: Current fuel after adding pump = 7
DEBUG: Fuel after covering distance = 4
DEBUG: Enqueued: petrol = 7, distance = 3 at position 5
DEBUG: Dequeued: petrol = 4, distance = 5 from position 4
DEBUG: Current fuel after adding pump = 8
DEBUG: Fuel after covering distance = 3
DEBUG: Enqueued: petrol = 4, distance = 5 at position 6
DEBUG: Dequeued: petrol = 6, distance = 5 from position 5
DEBUG: Current fuel after adding pump = 9
DEBUG: Fuel after covering distance = 4
DEBUG: Enqueued: petrol = 6, distance = 5 at position 7
DEBUG: Dequeued: petrol = 7, distance = 3 from position 6
DEBUG: Current fuel after adding pump = 11
DEBUG: Fuel after covering distance = 8
DEBUG: Enqueued: petrol = 7, distance = 3 at position 8
DEBUG: Tour is possible starting from index 2
Starting pump = 1
Tour: 1→2→3→0→1
DEBUG: Dequeued: petrol = 4, distance = 5 from position 7
DEBUG: Enqueued: petrol = 4, distance = 5 at position 9
DEBUG: Checking tour possibility starting at index 7
DEBUG: Dequeued: petrol = 6, distance = 5 from position 8
DEBUG: Current fuel after adding pump = 6
DEBUG: Fuel after covering distance = 1
DEBUG: Enqueued: petrol = 6, distance = 5 at position 10
DEBUG: Dequeued: petrol = 7, distance = 3 from position 9
DEBUG: Current fuel after adding pump = 8
DEBUG: Fuel after covering distance = 5
DEBUG: Enqueued: petrol = 7, distance = 3 at position 11
DEBUG: Dequeued: petrol = 4, distance = 5 from position 10
DEBUG: Current fuel after adding pump = 9
DEBUG: Fuel after covering distance = 4
DEBUG: Enqueued: petrol = 4, distance = 5 at position 12
DEBUG: Dequeued: petrol = 6, distance = 5 from position 11
DEBUG: Current fuel after adding pump = 10
DEBUG: Fuel after covering distance = 5
DEBUG: Enqueued: petrol = 6, distance = 5 at position 13
DEBUG: Tour is possible starting from index 7
Starting pump = 2
Tour: 2→3→0→1→2
DEBUG: Dequeued: petrol = 7, distance = 3 from position 12
DEBUG: Enqueued: petrol = 7, distance = 3 at position 14
DEBUG: Checking tour possibility starting at index 12
DEBUG: Dequeued: petrol = 4, distance = 5 from position 13
DEBUG: Current fuel after adding pump = 4
DEBUG: Not enough fuel to cover distance from pump (petrol = 4, distance = 5). Tour is impossible.
DEBUG: Dequeued: petrol = 6, distance = 5 from position 14
DEBUG: Enqueued: petrol = 6, distance = 5 at position 15

 ~/Desktop/College/Data Structures Sem 3/Experiment 2new  main !2
```

Output no 3: (debug =0)



Output no 4: diff testcase (`{{1, 5}, {2, 6}, {3, 7}};`

Another diff test case (`{{0, 0}, {0, 0}, {0, 0}};`)