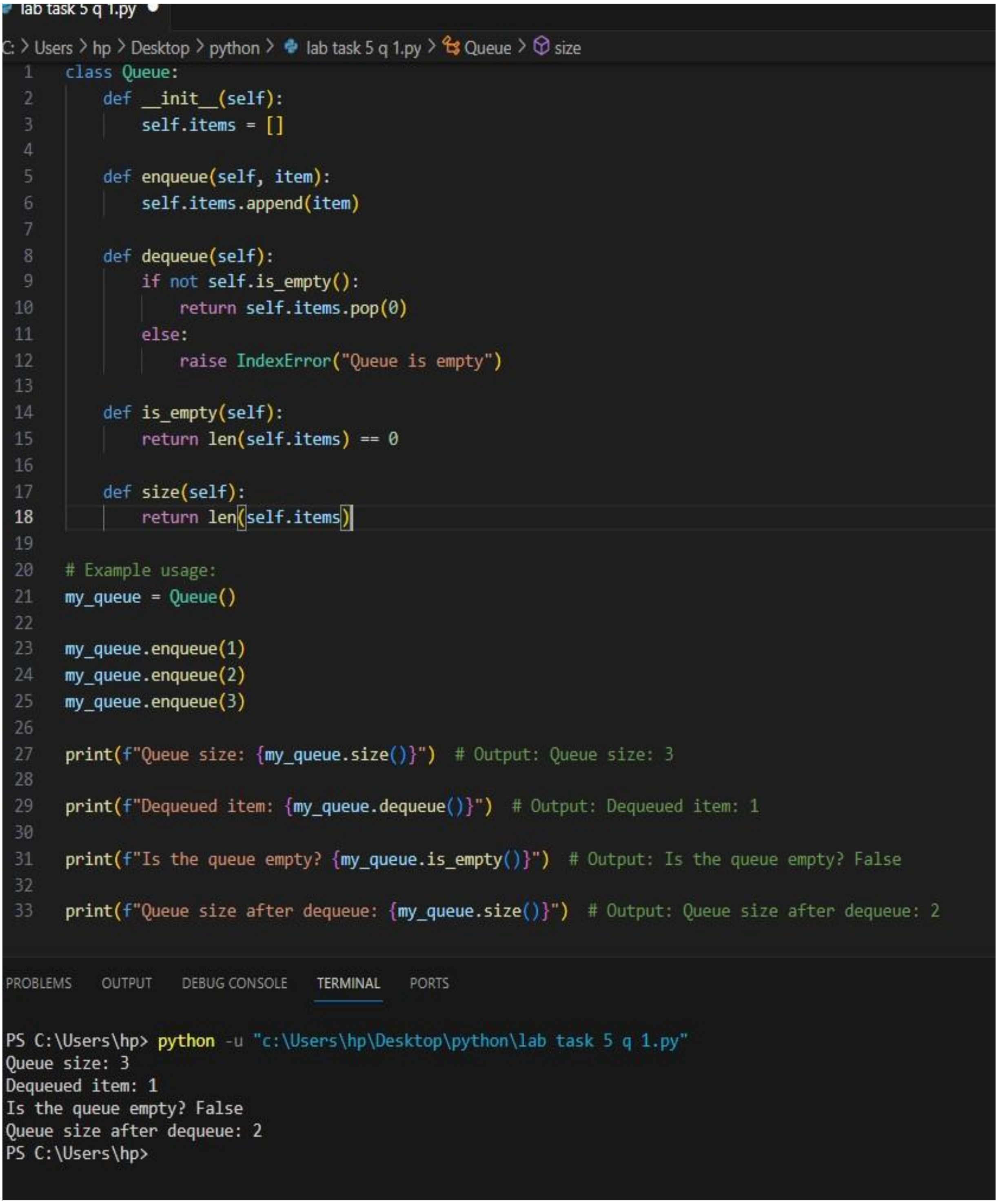
ABBOTTABAD UNIVERSITY OF SCIENCE AND TECHNOLOGY

NAME	SAAD SHAH
ROLL NO	12373
SUBJECT	DSA
LAB TASK	05
SUBMITTED TO	MR JAMAL ABDUL AHAD
DATED	11-11-2023

LAB TASK 05

Q NO 1: Implement a basic queue using an array in Python. Include methods for enqueue, dequeue, checking if the queue is empty, and finding the size of the queue.

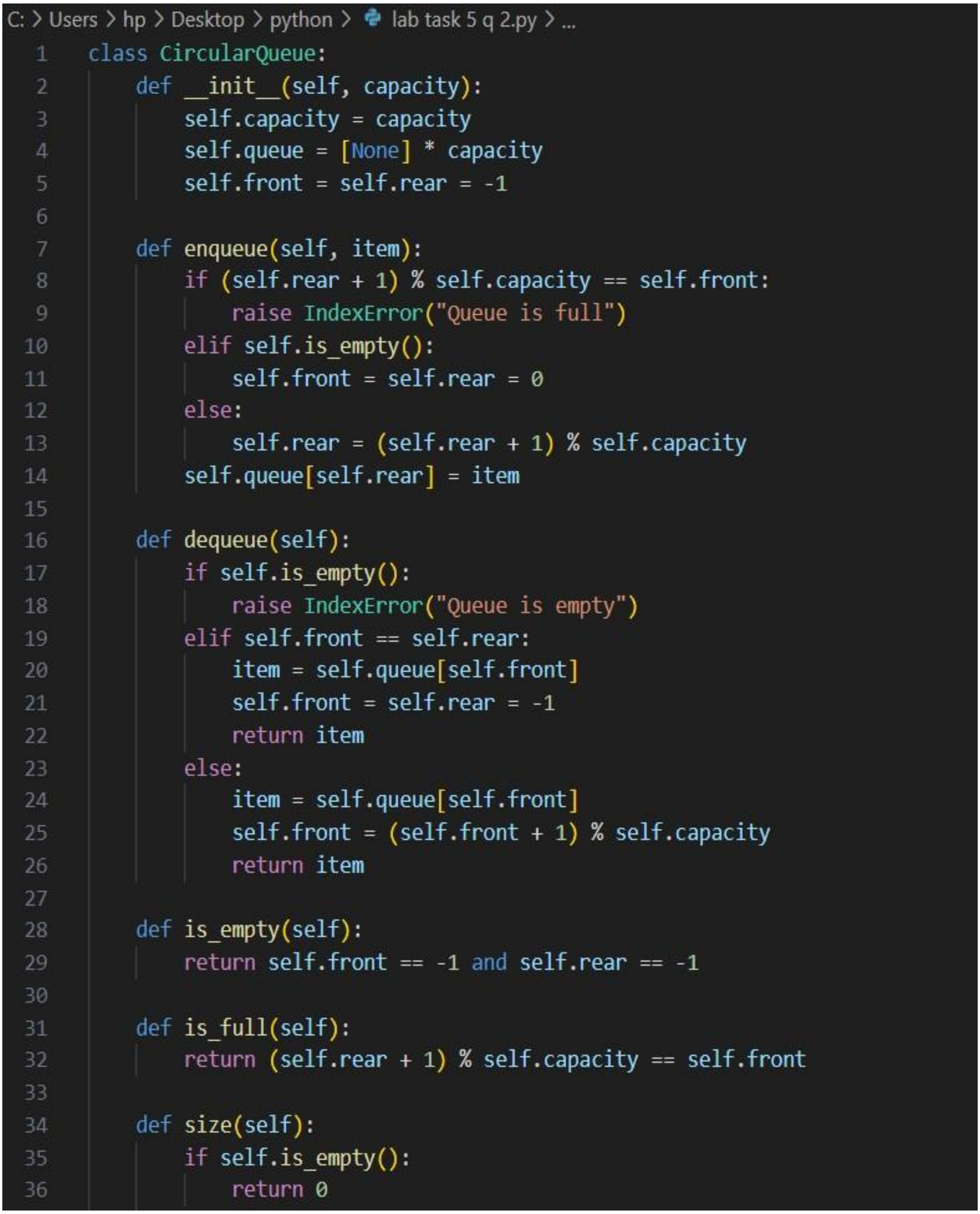


lab task 5 q 1.py
C: > Users > hp > Desktop > python > lab task 5 q 1.py > Queue > size

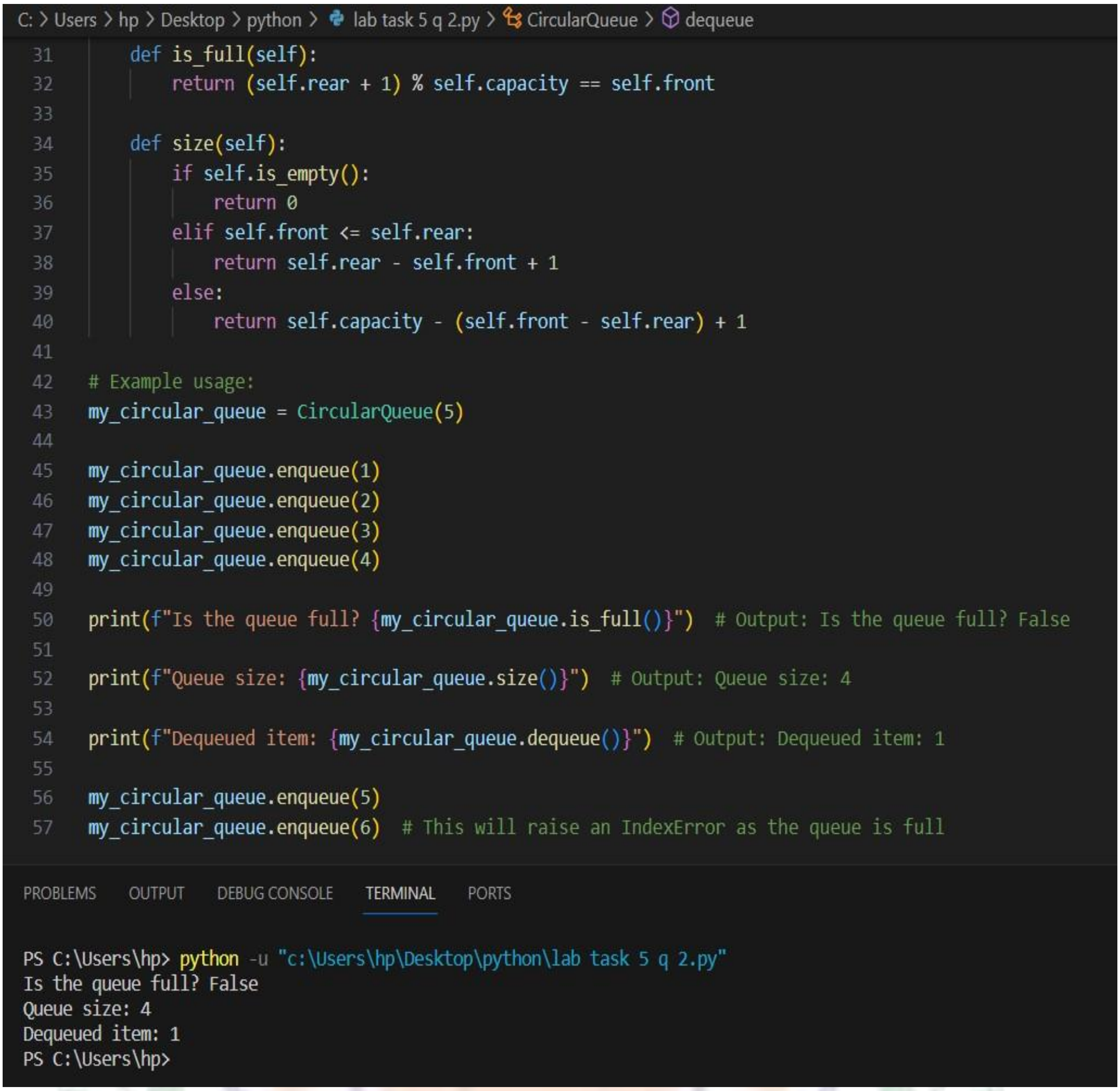
```python
class Queue:
    def __init__(self):
        self.items = []

    def enqueue(self, item):
        self.items.append(item)

    def dequeue(self):
        if not self.is_empty():
            return self.items.pop(0)
        else:
            raise IndexError("Queue is empty")

    def is_empty(self):
        return len(self.items) == 0

    def size(self):
        return len(self.items)

# Example usage:
my_queue = Queue()

my_queue.enqueue(1)
my_queue.enqueue(2)
my_queue.enqueue(3)

print(f"Queue size: {my_queue.size()}")  # Output: Queue size: 3

print(f"Dequeued item: {my_queue.dequeue()}")  # Output: Dequeued item: 1

print(f"Is the queue empty? {my_queue.is_empty()}")  # Output: Is the queue empty? False

print(f"Queue size after dequeue: {my_queue.size()}")  # Output: Queue size after dequeue: 2
```

```
PS C:\Users\hp> python -u "c:\Users\hp\Desktop\python\lab task 5 q 1.py"
Queue size: 3
Dequeued item: 1
Is the queue empty? False
Queue size after dequeue: 2
PS C:\Users\hp>
```

Q NO 2 : Implement a circular queue in Python. Include methods for enqueue, dequeue, checking if the queue is empty, checking if the queue is full, and finding the size of the queue.


C: > Users > hp > Desktop > python > lab task 5 q 2.py > ...

```python
class CircularQueue:
    def __init__(self, capacity):
        self.capacity = capacity
        self.queue = [None] * capacity
        self.front = self.rear = -1

    def enqueue(self, item):
        if (self.rear + 1) % self.capacity == self.front:
            raise IndexError("Queue is full")
        elif self.is_empty():
            self.front = self.rear = 0
        else:
            self.rear = (self.rear + 1) % self.capacity
        self.queue[self.rear] = item

    def dequeue(self):
        if self.is_empty():
            raise IndexError("Queue is empty")
        elif self.front == self.rear:
            item = self.queue[self.front]
            self.front = self.rear = -1
            return item
        else:
            item = self.queue[self.front]
            self.front = (self.front + 1) % self.capacity
            return item

    def is_empty(self):
        return self.front == -1 and self.rear == -1

    def is_full(self):
        return (self.rear + 1) % self.capacity == self.front

    def size(self):
        if self.is_empty():
            return 0
```


C: > Users > hp > Desktop > python > lab task 5 q 2.py > CircularQueue > dequeue
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```python
    def is_full(self):
        return (self.rear + 1) % self.capacity == self.front

    def size(self):
        if self.is_empty():
            return 0
        elif self.front <= self.rear:
            return self.rear - self.front + 1
        else:
            return self.capacity - (self.front - self.rear) + 1

# Example usage:
my_circular_queue = CircularQueue(5)

my_circular_queue.enqueue(1)
my_circular_queue.enqueue(2)
my_circular_queue.enqueue(3)
my_circular_queue.enqueue(4)

print(f"Is the queue full? {my_circular_queue.is_full()}")  # Output: Is the queue full? False

print(f"Queue size: {my_circular_queue.size()}")  # Output: Queue size: 4

print(f"Dequeued item: {my_circular_queue.dequeue()}")  # Output: Dequeued item: 1

my_circular_queue.enqueue(5)
my_circular_queue.enqueue(6)  # This will raise an IndexError as the queue is full
```

```
PS C:\Users\hp> python -u "c:\Users\hp\Desktop\python\lab task 5 q 2.py"
Is the queue full? False
Queue size: 4
Dequeued item: 1
PS C:\Users\hp>
```

THE END