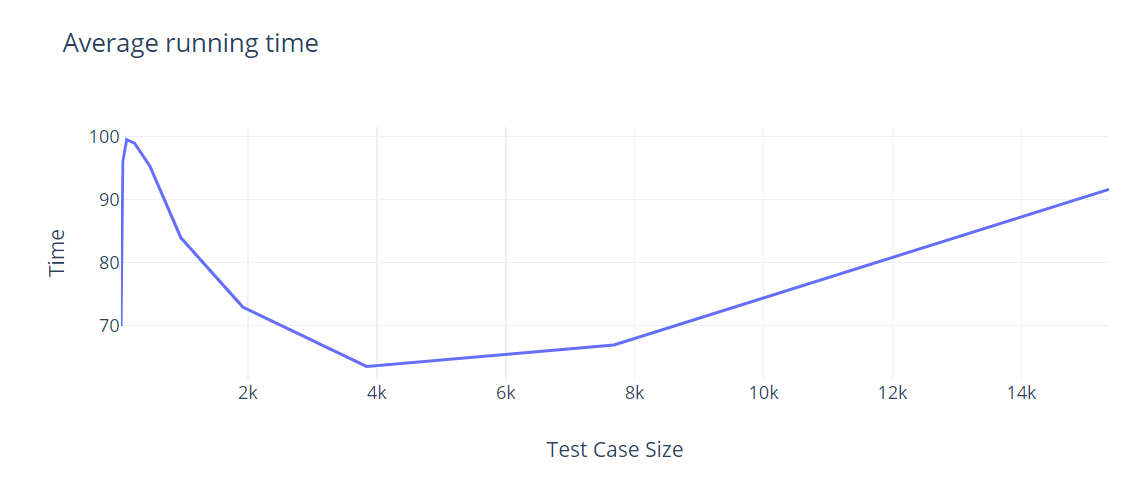
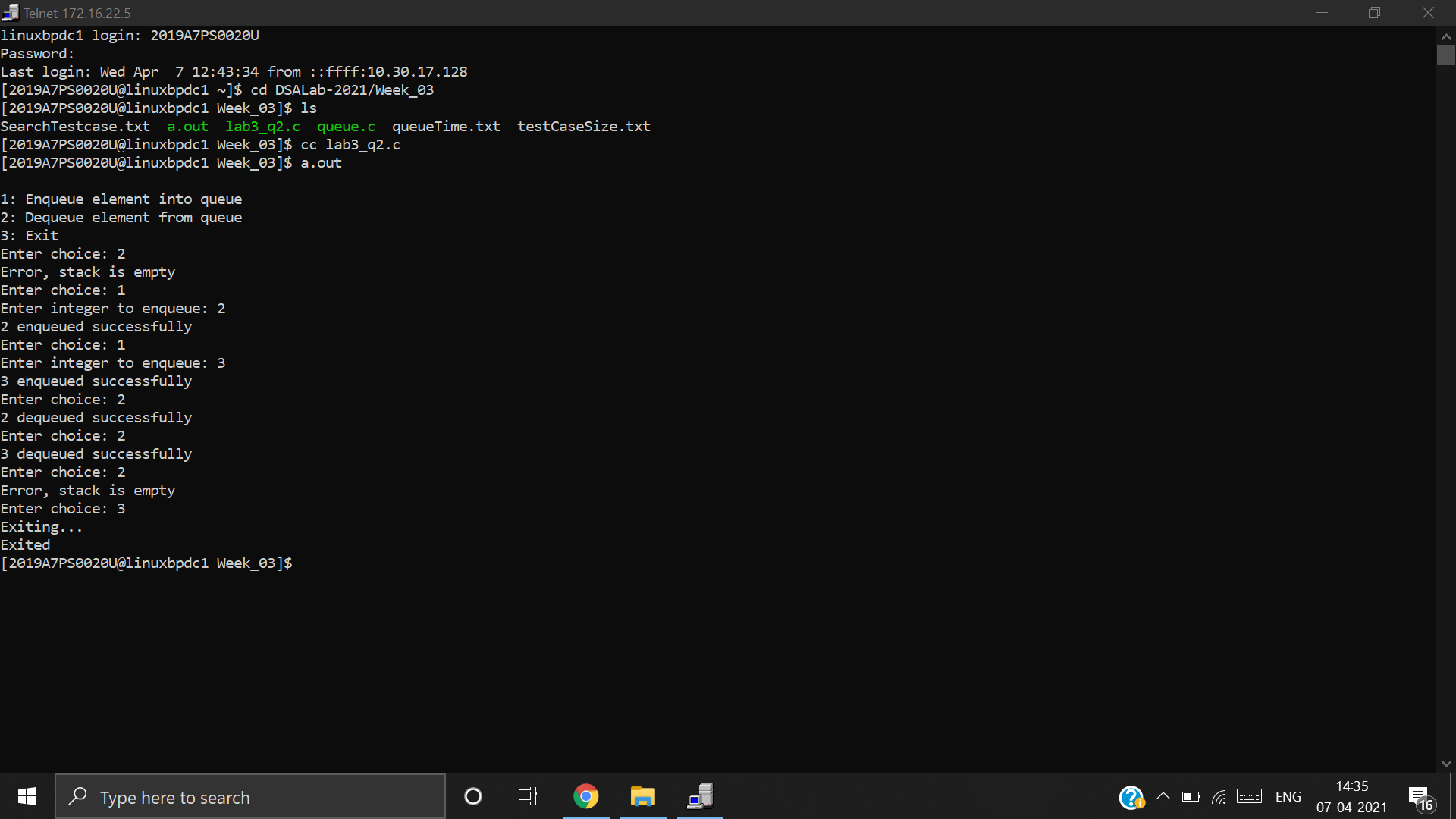
Lab 3

|  |  |
| --- | --- |
| Function | Big O |
|  |  |
| int\*queue;//the array that holds the queue  int queue\_size;//size of the queue[] array  int opCode;  int front=0, rear=0;//queue index variables | O(1) [declaration]  O(1) |
| int queueSize()  {  int size;  size=rear-front;  return size;  } | O(1)[declaration]  O(1)[arithmetic, assignment]  O(1)[return]  O(1) |
| int isQueueEmpty()  {  if (front==rear)  {return 1;}  else return 0;  } | O(1)[comparison]  O(1)[return]  O(1) |
| int isQueueFull()  {  if(rear==queue\_size)  {  return 1;  }  else return 0;  } | O(1)[comparison]  O(1)[return]  O(1) |
| int enQueue(int x)  {  if(rear==queue\_size)  {  return -1;  }  else {  queue[rear]=x;  rear=rear+1;  return 1;}  } | O(1)[comparison]  O(1)[return]  O(1)[index, assignment]  O(1)[arithmetic, assignment]  O(1) |
| int deQueue()  {  int x;  if(front==rear)  {  return -1;  }  else {  x=queue[front];  front=front++;  return x;}  } | O(1)[declaration]  O(1)[comparison]  O(1)[return]  O(1)[index, assignment]  O(1)[increment]  O(1) |



Ques 2:



|  |  |
| --- | --- |
| Function | Big O |
|  |  |
| int isStack2Full()  {  if(top2==stack\_size-1) return 1;  else return 0;  }  int isStack2Empty()//returns 1 if stack2[] is empty and 0 otherwise  {  if(top2==-1) return 1;  else return 0;  } | O(1)[comparison, return]  O(1)[comparison, return] |
| int pop2()  { int data;  if(!isStack2Empty())  {  data = stack2[top2];  top2 = top2 -1;  return data;  }  else  return -1;  }  int push2(int x)  {  if(!isStack2Full())  {  top2 = top2 + 1;  stack2[top2] = x;  return 1;  }  else  return -1;  } | O(1)[declaration]  O(1)[index, assignment]  O(1)[arithmetic, assignment]  O(1)[return]  O(1)[arithmetic, assignment]  O(1)[index, assignment]  O(1)[return]  **O(1)** |
| int deQueue()  {  if(isQueueEmpty())  {printf("Error, stack is empty");  return -1;}  else  {  while (isStack2Empty() == 0)  {  int x\_1=pop2();  push1(x\_1);  }  int y=pop1();  while (isStack1Empty() == 0)  {  int x\_2=pop1();  push2(x\_2);  }  printf("%d dequeued successfully",y);  return y;    }  } | O(n)[till end of list]  n  n  O(1)[pop func, assignment]  O(n)[till end of list]  n  n  **O(n)** |