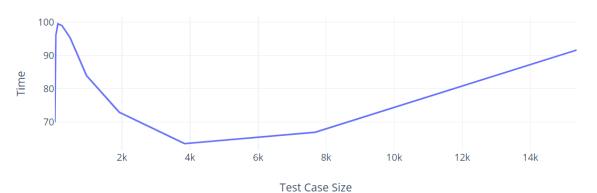
Lab 3

Function	Big O
Tanecian	5.6 0
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]
IIIt Iront=0, rear=0,//queue index variables	
int augus Cina/)	O(1)
int queueSize()	
{	0/4)[-
int size;	O(1)[declaration]
size=rear-front;	O(1)[arithmetic, assignment]
return size;	O(1)[return]
}	O(1)
int isQueueEmpty()	
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \	0(4)(
if (front==rear)	O(1)[comparison]
{return 1;}	O(1)[return]
else return 0;	-40
}	O(1)
int isQueueFull()	
{	
if(rear==queue_size)	O(1)[comparison]
{	
return 1;	O(1)[return]
}	
else return 0;	
}	O(1)
int enQueue(int x)	
{	
if(rear==queue_size)	O(1)[comparison]
{	
return -1;	O(1)[return]
}	
else {	
queue[rear]=x;	O(1)[index, assignment]
rear=rear+1;	O(1)[arithmetic, assignment]
return 1;}	
}	O(1)
int deQueue()	
{	
int x;	O(1)[declaration]
if(front==rear)	O(1)[comparison]
(
return -1;	O(1)[return]
}	
else {	
x=queue[front];	O(1)[index, assignment]
front=front++;	O(1)[increment]
return x;}	
. 550111 ///	

}	O(1)
j	0(1)

Average running time



Ques 2:

Function	Big O
int isStack2Full()	
{	
if(top2==stack_size-1) return 1;	
else return 0;	
}	O(1)[comparison, return]
int isStack2Empty()//returns 1 if stack2[] is	
empty and 0 otherwise	
{	
if(top2==-1) return 1;	
else return 0;	
}	O(1)[comparison, return]
int pop2()	
{ int data;	O(1)[declaration]
if(!isStack2Empty())	
{	

```
data = stack2[top2];
                                                   O(1)[index, assignment]
                                                   O(1)[arithmetic, assignment]
    top2 = top2 -1;
                                                   O(1)[return]
    return data;
 }
  else
    return -1;
}
int push2(int x)
 if(!isStack2Full())
                                                   O(1)[arithmetic, assignment]
    top2 = top2 + 1;
                                                   O(1)[index, assignment]
    stack2[top2] = x;
                                                   O(1)[return]
    return 1;
  }
  else
    return -1;
                                                  0(1)
int deQueue()
  if(isQueueEmpty())
  {printf("Error, stack is empty");
    return -1;}
  else
                                                   O(n)[till end of list]
    while (isStack2Empty() == 0)
      int x_1=pop2();
                                                  n
      push1(x_1);
                                                  n
    }
  int y=pop1();
                                                   O(1)[pop func, assignment]
  while (isStack1Empty() == 0)
                                                   O(n)[till end of list]
      int x_2=pop1();
                                                  n
      push2(x_2);
                                                  n
    printf("%d dequeued successfully",y);
  return y;
  }
                                                   O(n)
```