Name => Anmol Baranwal

Roll number => 2820208

Group => A3

Submitted to => Bhawna Ma'am

DSA LAB File

Question 1

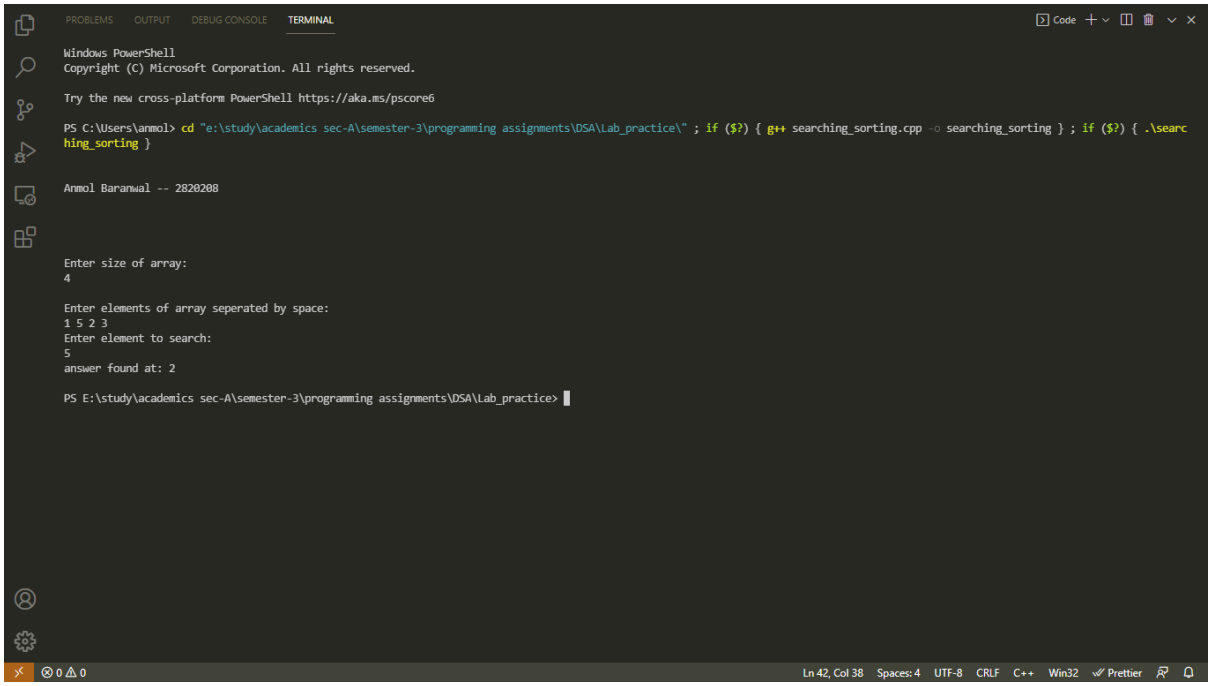
Write a program for Binary search method

Answer 1

```
#include <bits/stdc++.h>
    using namespace std;

    int main()
    {
        cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
```

```
int n, arr[50];
cout<<"\n\n";
cout<<"Enter size of array: \n";
cin>>n;
cout<<"\nEnter elements of array seperated by space:\n";
for(int i=0;i<n;i++){
   cin>> arr[i];
}

int key;
cout<<"Enter element to search:\n";
cin>>key;
int start=0,end=n-1,ans=-1;
while(start<=end){
   int mid = start + (end-start)/2;
   if(arr[mid]==key){
      ans=mid;
      break;

   } else if(arr[mid]>key){
      end=mid-1;
   } else{
      start=mid+1;
   }
}
if(ans==-1){
   cout<<"\nanswer not found:";
} else {
   cout<<"answer found at: "<<ans+1;
}
```

```
    cout<<"\n\n";

    return 0;
}
```



Question 2

Write a program for insertion sort, selection sort and bubble sort

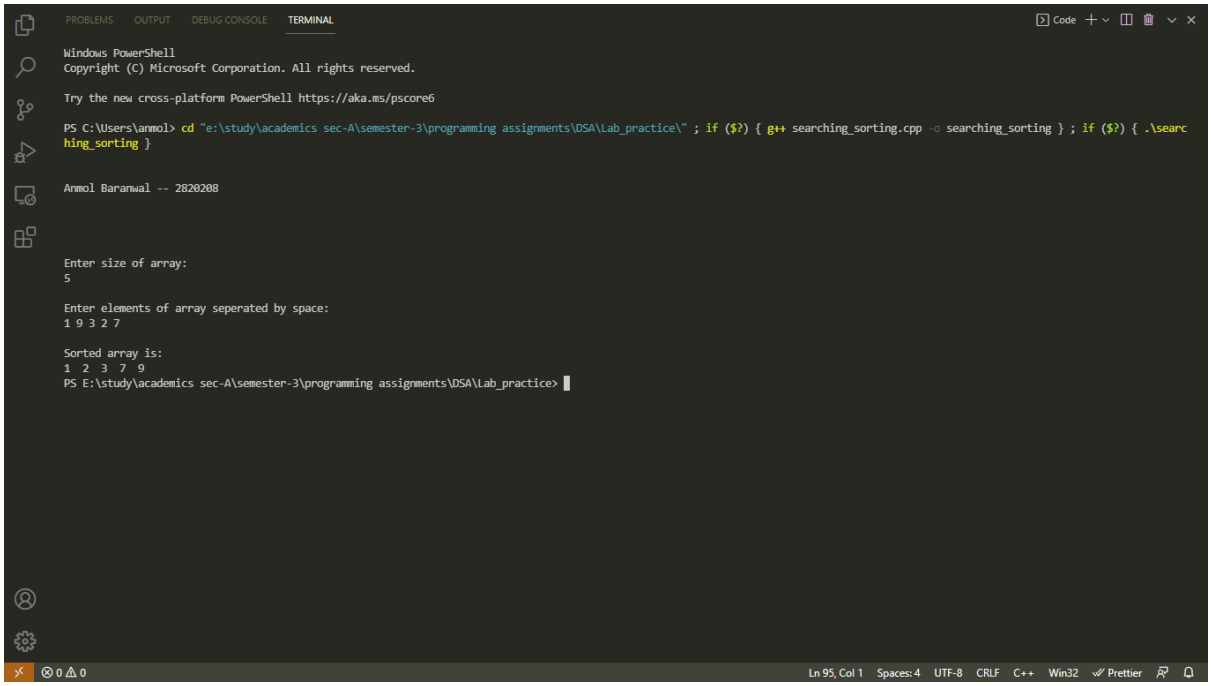
Answer 2

⇨ Bubble Sort

```
#include <bits/stdc++.h>
using namespace std;

    void displayArray(int arr[], int n){
```

```
    for(int i=0;i<n;i++){
        cout<<arr[i]<<"  ";
    }
}

void swapNo(int &x, int &y){
    int temp = x;
    x = y;
    y = temp;
}

int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    int n, arr[50];
    cout<<"\n\n";
    cout<<"Enter size of array: \n";
    cin>>n;
    cout<<"\nEnter elements of array seperated by space:\n";
    for(int i=0;i<n;i++){
        cin>> arr[i];
    }

    for(int i=0;i<=n-2;i++){
        int cnt=0;
        for(int j=0;j<=n-2-i;j++){
            if(arr[j]>arr[j+1]){
                cnt+=1;
                swap(arr[j],arr[j+1]);
            }
```

```
        }
        if(cnt==0)
          break;
    }
    cout<<"\nSorted array is:\n";
    displayArray(arr,n);

    return 0;
}
```

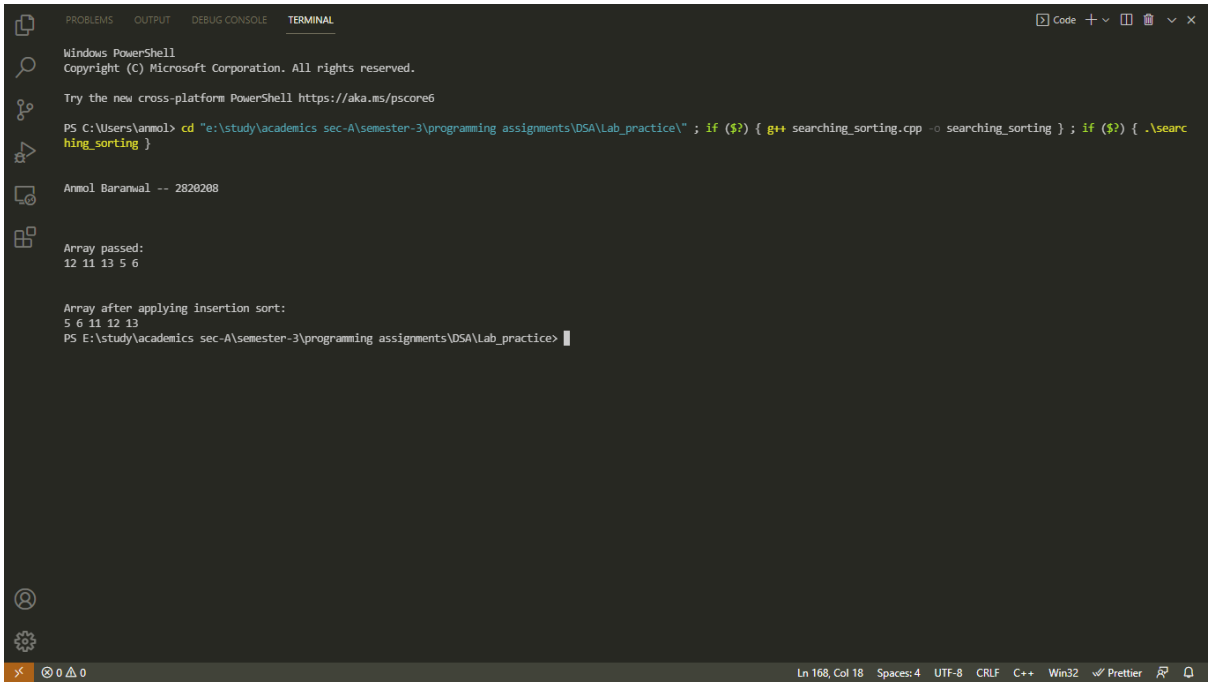


⇨ Insertion Sort

```
#include <bits/stdc++.h>
    using namespace std;

    /* Function to sort an array using insertion sort*/
    void insertionSort(int arr[], int n)
```

```
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;

        /* Move elements of arr[0..i-1], that are
        greater than key, to one position ahead
        of their current position */
        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

// A utility function to print an array of size n
void printArray(int arr[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}


int main()
```

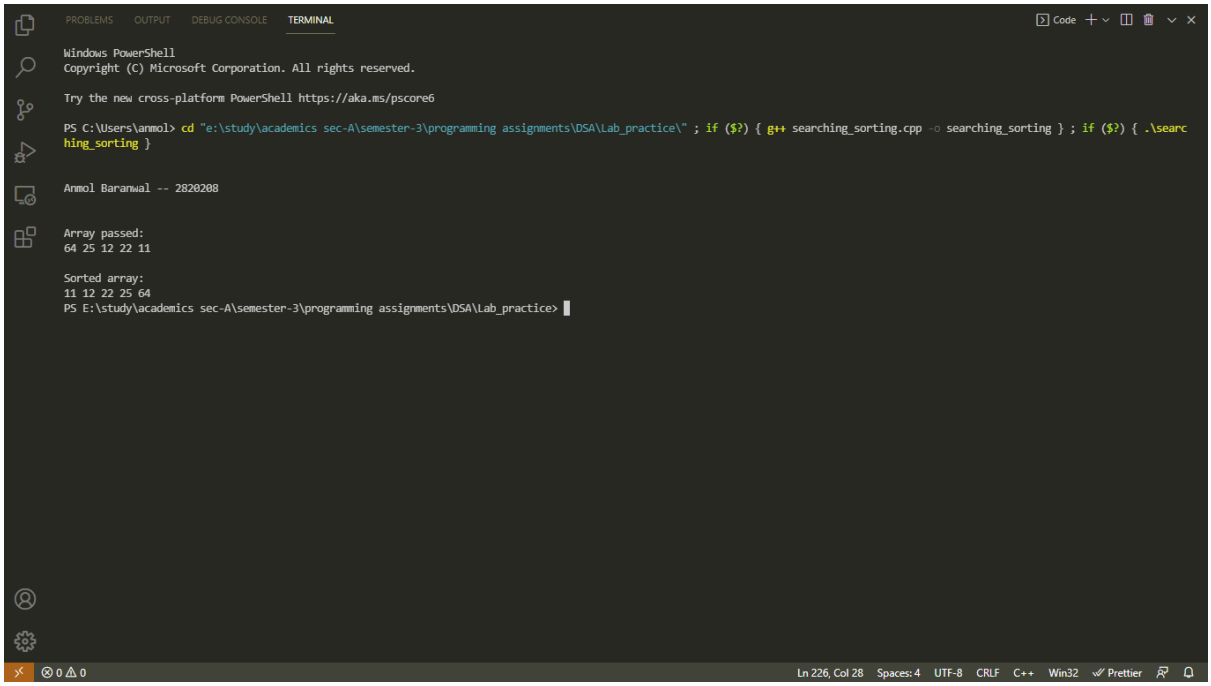
```
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    int arr[] = { 12, 11, 13, 5, 6 };
    int n = sizeof(arr) / sizeof(arr[0]);

    cout<<"\nArray passed: \n";
    printArray(arr,n);

    insertionSort(arr, n);
    cout<<"\n\nArray after applying insertion sort: \n";
    printArray(arr, n);

    return 0;
}
```



⇨ Selection Sort

```
#include <bits/stdc++.h>
   using namespace std;

   void swap(int *xp, int *yp)
   {
      int temp = *xp;
      *xp = *yp;
      *yp = temp;
   }

   void selectionSort(int arr[], int n)
   {
      int i, j, min_idx;

      // One by one move boundary of unsorted subarray
      for (i = 0; i < n-1; i++)
      {
         // Find the minimum element in unsorted array
         min_idx = i;
         for (j = i+1; j < n; j++)
         if (arr[j] < arr[min_idx])
            min_idx = j;

         // Swap the found minimum element with the first element
         swap(&arr[min_idx], &arr[i]);
      }
   }

   /* Function to print an array */
   void printArray(int arr[], int size)
```

```
{
    int i;
    for (i=0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}


int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr)/sizeof(arr[0]);

    cout<<"Array passed: \n";
    printArray(arr, n);
    selectionSort(arr, n);
    cout << "\nSorted array: \n";
    printArray(arr, n);
    return 0;
}
```



Question 3

Write a program to implement Stack and its operation

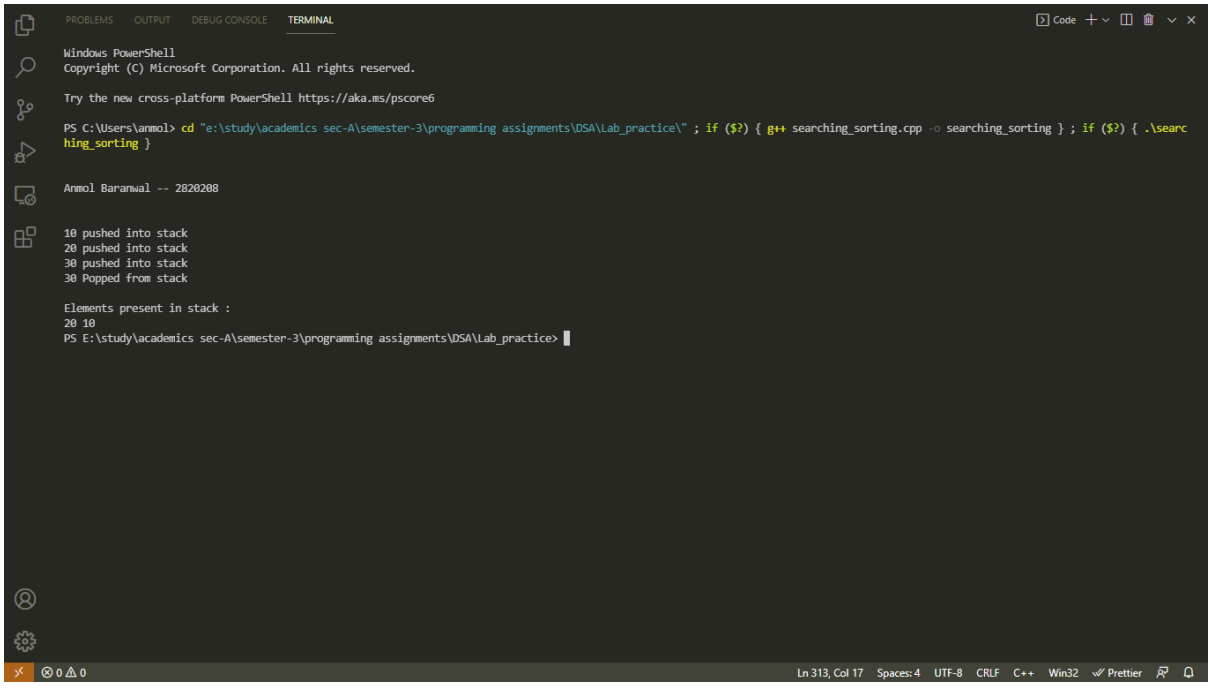
Answer 3

```
#include <bits/stdc++.h>

using namespace std;

#define MAX 1000

class Stack {
  int top;
```

```
public:
  int a[MAX]; // Maximum size of Stack

  Stack() { top = -1; }
  bool push(int x);
  int pop();
  int peek();
  bool isEmpty();
};

bool Stack::push(int x)
{
  if (top >= (MAX - 1)) {
    cout << "Stack Overflow";
    return false;
  }
  else {
    a[++top] = x;
    cout << x << " pushed into stack\n";
    return true;
  }
}

int Stack::pop()
{
  if (top < 0) {
    cout << "Stack Underflow";
    return 0;
  }
  else {
```

```
    int x = a[top--];
    return x;
  }
}
int Stack::peek()
{
  if (top < 0) {
    cout << "Stack is Empty";
    return 0;
  }
  else {
    int x = a[top];
    return x;
  }
}

bool Stack::isEmpty()
{
  return (top < 0);
}

// Driver program to test above functions
int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
  class Stack s;
  s.push(10);
  s.push(20);
  s.push(30);
  cout << s.pop() << " Popped from stack\n";
```

```
 //print all elements in stack :
 cout<<"\nElements present in stack : \n";
 while(!s.isEmpty())
 {
   // print top element in stack
   cout<<s.peek()<<" ";
   // remove top element in stack
   s.pop();
 }

 return 0;
}
```



Question 4

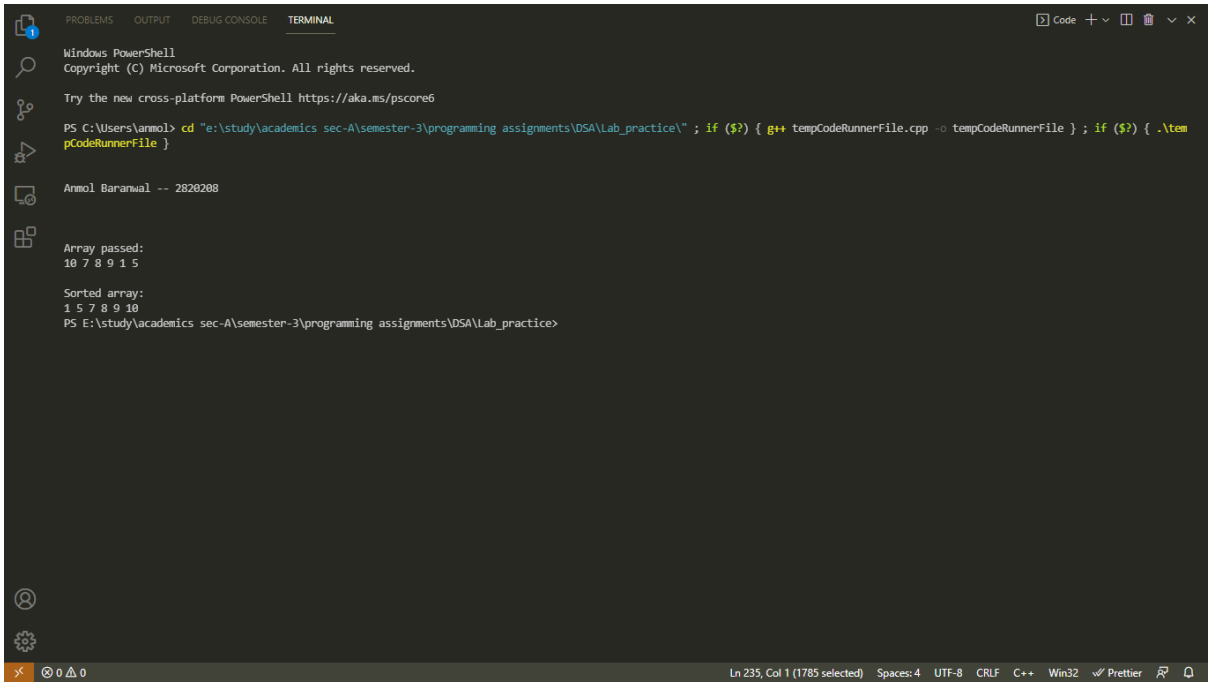
Write a program for quick sort

Answer 4

```
#include <bits/stdc++.h>
   using namespace std;


   // A utility function to swap two elements
   void swap(int* a, int* b)
   {
      int t = *a;
      *a = *b;
      *b = t;
   }


   int partition (int arr[], int low, int high)
   {
      int pivot = arr[high]; // pivot
      int i = (low - 1); // Index of smaller element and indicates the right position of pivot found so far


      for (int j = low; j <= high - 1; j++)
      {
         // If current element is smaller than the pivot
         if (arr[j] < pivot)
         {
            i++; // increment index of smaller element
            swap(&arr[i], &arr[j]);
         }
      }
      swap(&arr[i + 1], &arr[high]);
      return (i + 1);
   }
```

```
void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {
        /* pi is partitioning index, arr[p] is now
        at right place */
        int pi = partition(arr, low, high);

        // Separately sort elements before
        // partition and after partition
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

/* Function to print an array */
void printArray(int arr[], int size)
{
    int i;
    for (i = 0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    int arr[] = {10, 7, 8, 9, 1, 5};
    int n = sizeof(arr) / sizeof(arr[0]);
    cout<<"\nArray passed: \n";
```

```
    printArray(arr, n);
    quickSort(arr, 0, n - 1);
    cout << "\nSorted array: \n";
    printArray(arr, n);
    return 0;
}
```



Question 5

Write a program for merge sort

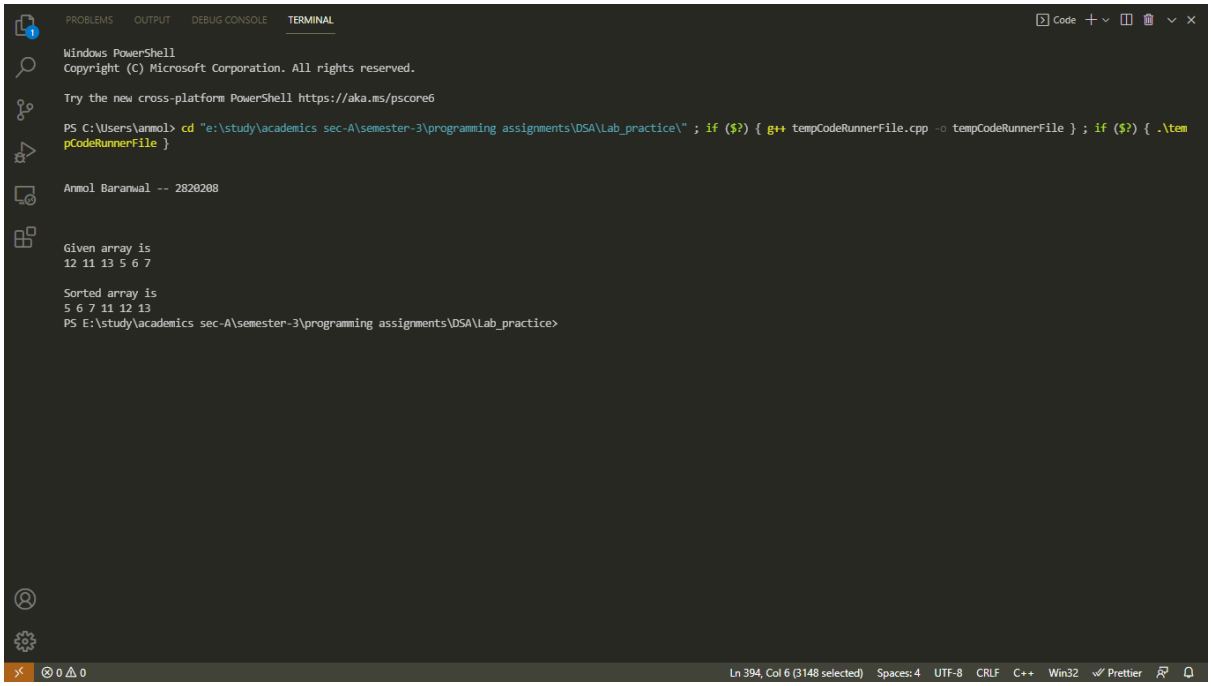
Answer 5

```
#include <bits/stdc++.h>
    using namespace std;

    // Merges two subarrays of array[].
```

```
// First subarray is arr[begin..mid]
// Second subarray is arr[mid+1..end]
void merge(int array[], int const left, int const mid, int const right)
{
    auto const subArrayOne = mid - left + 1;
    auto const subArrayTwo = right - mid;

    // Create temp arrays
    auto *leftArray = new int[subArrayOne],
        *rightArray = new int[subArrayTwo];

    // Copy data to temp arrays leftArray[] and rightArray[]
    for (auto i = 0; i < subArrayOne; i++)
        leftArray[i] = array[left + i];
    for (auto j = 0; j < subArrayTwo; j++)
        rightArray[j] = array[mid + 1 + j];

    auto indexOfSubArrayOne = 0, // Initial index of first sub-array
        indexOfSubArrayTwo = 0; // Initial index of second sub-array
    int indexOfMergedArray = left; // Initial index of merged array

    // Merge the temp arrays back into array[left..right]
    while (indexOfSubArrayOne < subArrayOne && indexOfSubArrayTwo <
subArrayTwo) {
        if (leftArray[indexOfSubArrayOne] <= rightArray[indexOfSubArrayTwo]) {
            array[indexOfMergedArray] = leftArray[indexOfSubArrayOne];
            indexOfSubArrayOne++;
        }
        else {
            array[indexOfMergedArray] = rightArray[indexOfSubArrayTwo];
            indexOfSubArrayTwo++;
```

```
        }
        indexOfMergedArray++;
    }
    // Copy the remaining elements of
    // left[], if there are any
    while (indexOfSubArrayOne < subArrayOne) {
        array[indexOfMergedArray] = leftArray[indexOfSubArrayOne];
        indexOfSubArrayOne++;
        indexOfMergedArray++;
    }
    // Copy the remaining elements of
    // right[], if there are any
    while (indexOfSubArrayTwo < subArrayTwo) {
        array[indexOfMergedArray] = rightArray[indexOfSubArrayTwo];
        indexOfSubArrayTwo++;
        indexOfMergedArray++;
    }
}

// begin is for left index and end is
// right index of the sub-array
// of arr to be sorted */
void mergeSort(int array[], int const begin, int const end)
{
    if (begin >= end)
        return; // Returns recursively

    auto mid = begin + (end - begin) / 2;
    mergeSort(array, begin, mid);
    mergeSort(array, mid + 1, end);
```

```
    merge(array, begin, mid, end);
}

// UTILITY FUNCTIONS
// Function to print an array
void printArray(int A[], int size)
{
    for (auto i = 0; i < size; i++)
        cout << A[i] << " ";
}

int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    int arr[] = { 12, 11, 13, 5, 6, 7 };
    auto arr_size = sizeof(arr) / sizeof(arr[0]);

    cout << "\nGiven array is \n";
    printArray(arr, arr_size);

    mergeSort(arr, 0, arr_size - 1);

    cout << "\n\nSorted array is \n";
    printArray(arr, arr_size);
    return 0;
}
```



Question 6

Write a program to implement Queue and its operation

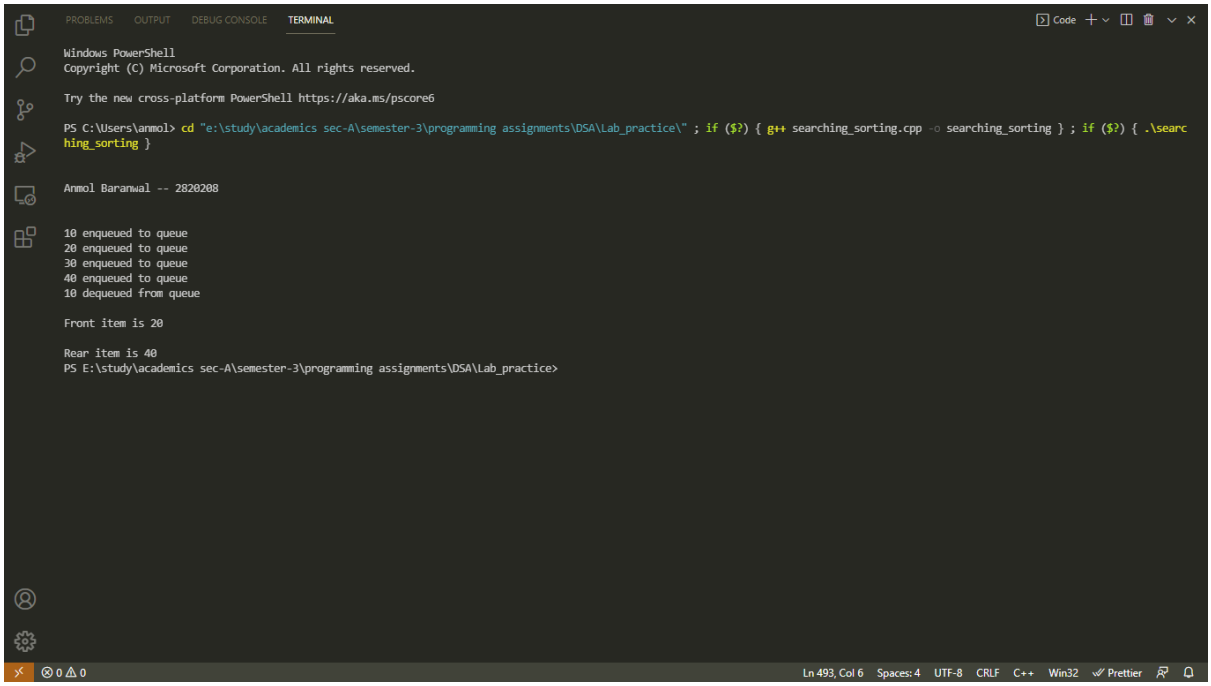
Answer 6

```
#include <bits/stdc++.h>
   using namespace std;

   // A structure to represent a queue
   class Queue {
   public:
      int front, rear, size;
      unsigned capacity;
      int* array;
   };
```

```
Queue* createQueue(unsigned capacity)
{
    Queue* queue = new Queue();
    queue->capacity = capacity;
    queue->front = queue->size = 0;

    // This is important, see the enqueue
    queue->rear = capacity - 1;
    queue->array = new int[queue->capacity];
    return queue;
}

int isFull(Queue* queue)
{
    return (queue->size == queue->capacity);
}

// Queue is empty when size is 0
int isEmpty(Queue* queue)
{
    return (queue->size == 0);
}

// It changes rear and size
void enqueue(Queue* queue, int item)
{
    if (isFull(queue))
        return;
    queue->rear = (queue->rear + 1)
            % queue->capacity;
```

```
    queue->array[queue->rear] = item;
    queue->size = queue->size + 1;
    cout << item << " enqueued to queue\n";
}

// It changes front and size
int dequeue(Queue* queue)
{
    if (isEmpty(queue))
        return INT_MIN;
    int item = queue->array[queue->front];
    queue->front = (queue->front + 1)
                % queue->capacity;
    queue->size = queue->size - 1;
    return item;
}

// Function to get front of queue
int front(Queue* queue)
{
    if (isEmpty(queue))
        return INT_MIN;
    return queue->array[queue->front];
}

// Function to get rear of queue
int rear(Queue* queue)
{
    if (isEmpty(queue))
        return INT_MIN;
```

```
    return queue->array[queue->rear];
}

int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    Queue* queue = createQueue(1000);

    enqueue(queue, 10);
    enqueue(queue, 20);
    enqueue(queue, 30);
    enqueue(queue, 40);

    cout << dequeue(queue)
        << " dequeued from queue\n";

    cout << "\nFront item is "
        << front(queue) << endl;
    cout << "\nRear item is "
        << rear(queue) << endl;

    return 0;
}
```



Question 7

Write a program to implement Circular Queue and its operation

Answer 7

```
#include<bits/stdc++.h>
   using namespace std;

   class Queue
   {
      // Initialize front and rear
      int rear, front;
```

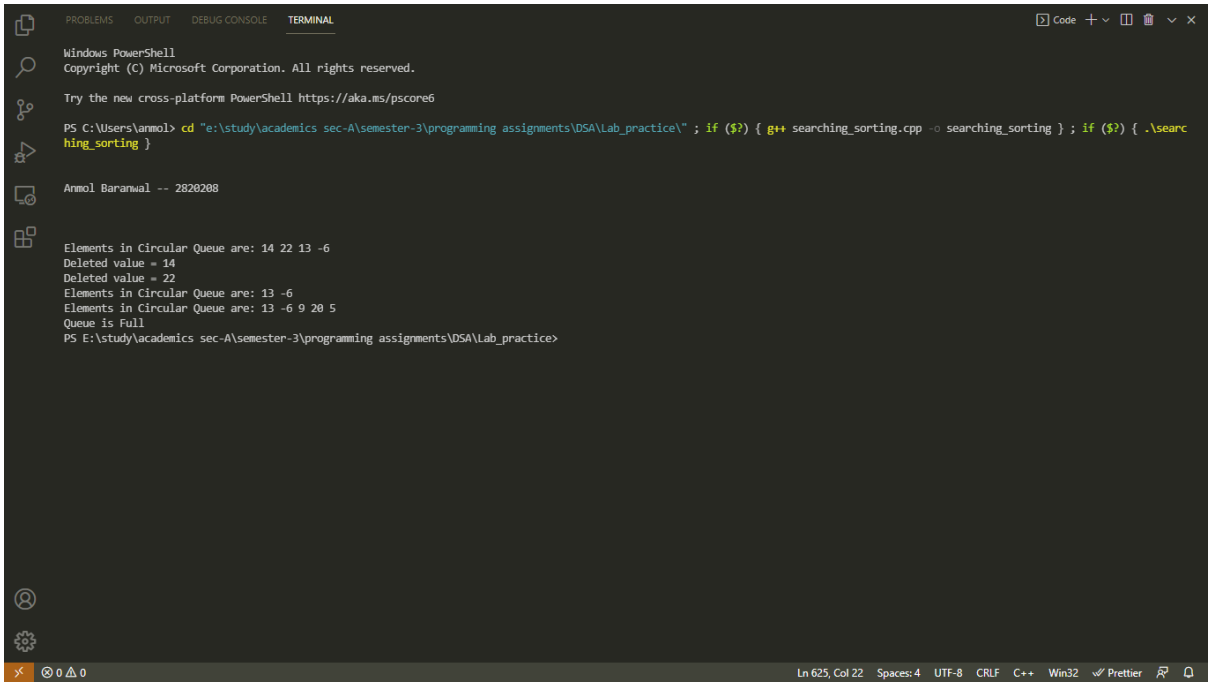
```
    // Circular Queue
    int size;
    int *arr;
public:
    Queue(int s)
    {
    front = rear = -1;
    size = s;
    arr = new int[s];
    }

    void enQueue(int value);
    int deQueue();
    void displayQueue();
};


/* Function to create Circular queue */
void Queue::enQueue(int value)
{
    if ((front == 0 && rear == size-1) ||
            (rear == (front-1)%(size-1)))
    {
        printf("\nQueue is Full");
        return;
    }

    else if (front == -1) /* Insert First Element */
    {
        front = rear = 0;
```

```
        arr[rear] = value;
    }

    else if (rear == size-1 && front != 0)
    {
        rear = 0;
        arr[rear] = value;
    }

    else
    {
        rear++;
        arr[rear] = value;
    }
}

// Function to delete element from Circular Queue
int Queue::deQueue()
{
    if (front == -1)
    {
        printf("\nQueue is Empty");
        return INT_MIN;
    }

    int data = arr[front];
    arr[front] = -1;
    if (front == rear)
    {
        front = -1;
```

```
        rear = -1;
    }
    else if (front == size-1)
        front = 0;
    else
        front++;

    return data;
}

// Function displaying the elements
// of Circular Queue
void Queue::displayQueue()
{
    if (front == -1)
    {
        printf("\nQueue is Empty");
        return;
    }
    printf("\nElements in Circular Queue are: ");
    if (rear >= front)
    {
        for (int i = front; i <= rear; i++)
            printf("%d ",arr[i]);
    }
    else
    {
        for (int i = front; i < size; i++)
            printf("%d ", arr[i]);
```

```
        for (int i = 0; i <= rear; i++)
            printf("%d ", arr[i]);
    }
}

/* Driver of the program */
int main()
{
    cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
    Queue q(5);

    // Inserting elements in Circular Queue
    q.enQueue(14);
    q.enQueue(22);
    q.enQueue(13);
    q.enQueue(-6);

    // Display elements present in Circular Queue
    q.displayQueue();

    // Deleting elements from Circular Queue
    printf("\nDeleted value = %d", q.deQueue());
    printf("\nDeleted value = %d", q.deQueue());

    q.displayQueue();

    q.enQueue(9);
    q.enQueue(20);
    q.enQueue(5);
```

```
    q.displayQueue();

    q.enQueue(20);
    return 0;
}
```



Question 8

Write a program to implement singly linked list for the following operations: Create, Display, searching, traversing and deletion

Answer 8

```
#include <stdio.h>
   #include<stdlib.h>
```

```
// Linked List Node
struct node {
    int info;
    struct node* link;
};
struct node* start = NULL;

// Function to traverse the linked list
void traverse()
{
    struct node* temp;

    // List is empty
    if (start == NULL)
        printf("\nList is empty\n");

    // Else print the LL
    else {
        temp = start;
        while (temp != NULL) {
            printf("Data = %d\n",
                temp->info);
            temp = temp->link;
        }
    }
}

// Function to insert at the front
// of the linked list
void insertAtFront()
```

```
{
    int data;
    struct node* temp;
    temp = malloc(sizeof(struct node));
    printf("\nEnter number to"
        " be inserted : ");
    scanf("%d", &data);
    temp->info = data;

    // Pointer of temp will be
    // assigned to start
    temp->link = start;
    start = temp;
}

// Function to insert at the end of
// the linked list
void insertAtEnd()
{
    int data;
    struct node *temp, *head;
    temp = malloc(sizeof(struct node));

    // Enter the number
    printf("\nEnter number to"
        " be inserted : ");
    scanf("%d", &data);

    // Changes links
    temp->link = 0;
```

```
    temp->info = data;
    head = start;
    while (head->link != NULL) {
        head = head->link;
    }
    head->link = temp;
}

// Function to insert at any specified
// position in the linked list
void insertAtPosition()
{
    struct node *temp, *newnode;
    int pos, data, i = 1;
    newnode = malloc(sizeof(struct node));

    // Enter the position and data
    printf("\nEnter position and data :");
    scanf("%d %d", &pos, &data);

    // Change Links
    temp = start;
    newnode->info = data;
    newnode->link = 0;
    while (i < pos - 1) {
        temp = temp->link;
        i++;
    }
    newnode->link = temp->link;
    temp->link = newnode;
```

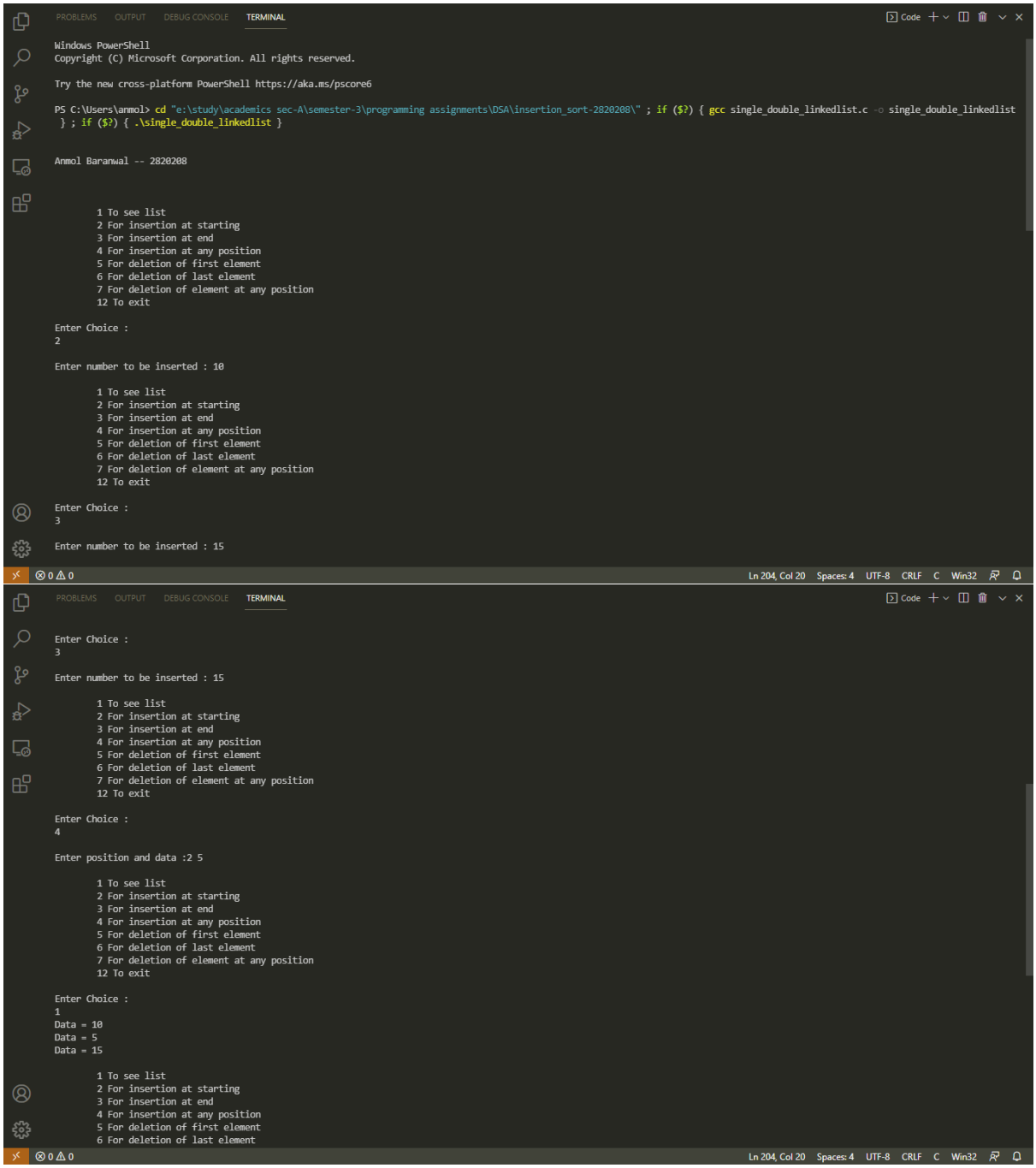
```
}

// Function to delete from the front
// of the linked list
void deleteFirst()
{
    struct node* temp;
    if (start == NULL)
        printf("\nList is empty\n");
    else {
        temp = start;
        start = start->link;
        free(temp);
    }
}

// Function to delete from the end
// of the linked list
void deleteEnd()
{
    struct node *temp, *prevnode;
    if (start == NULL)
        printf("\nList is Empty\n");
    else {
        temp = start;
        while (temp->link != 0) {
            prevnode = temp;
            temp = temp->link;
        }
        free(temp);
```

```
        prevnode->link = 0;
    }
}

// Function to delete from any specified
// position from the linked list
void deletePosition()
{
    struct node *temp, *position;
    int i = 1, pos;

    // If LL is empty
    if (start == NULL)
        printf("\nList is empty\n");

    // Otherwise
    else {
        printf("\nEnter index : ");

        // Position to be deleted
        scanf("%d", &pos);
        position = malloc(sizeof(struct node));
        temp = start;

        // Traverse till position
        while (i < pos - 1) {
            temp = temp->link;
            i++;
        }
```

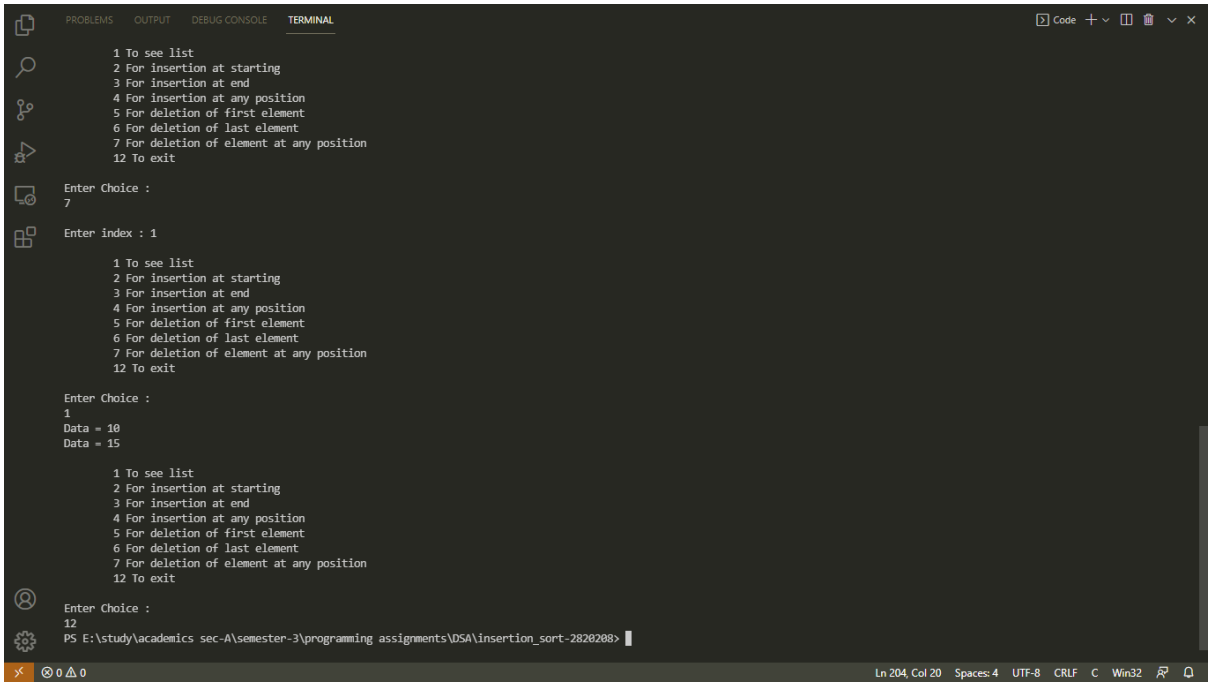
```
        // Change Links
        position = temp->link;
        temp->link = position->link;

        // Free memory
        free(position);
    }
}

// Driver Code
int main()
{
    printf("\n\nAnmol Baranwal -- 2820208 \n\n\n");
    int choice;
    while (1) {

        printf("\n\t1 To see list\n");
        printf("\t2 For insertion at"
            " starting\n");
        printf("\t3 For insertion at"
            " end\n");
        printf("\t4 For insertion at "
            "any position\n");
        printf("\t5 For deletion of "
            "first element\n");
        printf("\t6 For deletion of "
            "last element\n");
        printf("\t7 For deletion of "
            "element at any position\n");
        printf("\t12 To exit\n");
```

```
printf("\nEnter Choice :\n");
scanf("%d", &choice);

switch (choice) {
case 1:
   traverse();
   break;
case 2:
   insertAtFront();
   break;
case 3:
   insertAtEnd();
   break;
case 4:
   insertAtPosition();
   break;
case 5:
   deleteFirst();
   break;
case 6:
   deleteEnd();
   break;
case 7:
   deletePosition();
   break;
case 12:
   exit(1);
   break;
default:
   printf("Incorrect Choice\n");
```

```
        }
    }
    return 0;
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\anmol> cd "e:\study\academics sec-A\semester-3\programming assignments\DSA\insertion_sort-2820208\" ; if ($?) { gcc single_double_linkedlist.c -o single_double_linkedlist } ; if ($?) { .\single_double_linkedlist }
Anmol Baranwal -- 2820208
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
12 To exit
Enter Choice :
2
Enter number to be inserted : 10
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
12 To exit
Enter Choice :
3
Enter number to be inserted : 15
0 0
Ln 204, Col 20
Spaces: 4
UTF-8
CRLF
C
Win32

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
Enter Choice :
3
Enter number to be inserted : 15
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
12 To exit
Enter Choice :
4
Enter position and data :2 5
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
12 To exit
Enter Choice :
1
Data = 10
Data = 5
Data = 15
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
0 0
Ln 204, Col 20
Spaces: 4
UTF-8
CRLF
C
Win32



Question 9

Write a program to implement doubly linked list for the following operations: Create, Display, inserting, counting, searching, traversing and deletion

Answer 9

```
#include <stdio.h>
#include <stdlib.h>

// Linked List Node
struct node {
        int info;
        struct node *prev, *next;
};
struct node* start = NULL;
```

```
// Function to traverse the linked list
void traverse()
{
	// List is empty
	if (start == NULL) {
		printf("\nList is empty\n");
		return;
	}
	// Else print the Data
	struct node* temp;
	temp = start;
	while (temp != NULL) {
		printf("Data = %d\n", temp->info);
		temp = temp->next;
	}
}

// Function to insert at the front
// of the linked list
void insertAtFront()
{
	int data;
	struct node* temp;
	temp = (struct node*)malloc(sizeof(struct node));
	printf("\nEnter number to be inserted: ");
	scanf("%d", &data);
	temp->info = data;
	temp->prev = NULL;

	// Pointer of temp will be
```

```
        // assigned to start
        temp->next = start;
        start = temp;
}

// Function to insert at the end of
// the linked list
void insertAtEnd()
{
        int data;
        struct node *temp, *trav;
        temp = (struct node*)malloc(sizeof(struct node));
        temp->prev = NULL;
        temp->next = NULL;
        printf("\nEnter number to be inserted: ");
        scanf("%d", &data);
        temp->info = data;
        temp->next = NULL;
        trav = start;

        // If start is NULL
        if (start == NULL) {

                start = temp;
        }

        // Changes Links
        else {
                while (trav->next != NULL)
                        trav = trav->next;
```

```
                temp->prev = trav;
                trav->next = temp;
        }
}

// Function to insert at any specified
// position in the linked list
void insertAtPosition()
{
        int data, pos, i = 1;
        struct node *temp, *newnode;
        newnode = malloc(sizeof(struct node));
        newnode->next = NULL;
        newnode->prev = NULL;

        // Enter the position and data
        printf("\nEnter position : ");
        scanf("%d", &pos);



        // If start==NULL,
        if (start == NULL) {
                start = newnode;
                newnode->prev = NULL;
                newnode->next = NULL;
        }

        // If position==1,
        else if (pos == 1) {
        // this is author method its correct but we can simply call insertAtfront() function
for this special case
```

```
        /* newnode->next = start;
                newnode->next->prev = newnode;
                newnode->prev = NULL;
                start = newnode; */
        // now this is improved by Jay Ghughriwala on geeksforgeeks
        insertAtFront();
        }

        // Change links
        else {
        printf("\nEnter number to be inserted: ");
        scanf("%d", &data);
        newnode->info = data;
        temp = start;
                while (i < pos - 1) {
                        temp = temp->next;
                        i++;
                }
                newnode->next = temp->next;
                newnode->prev = temp;
                temp->next = newnode;
                temp->next->prev = newnode;
        }
}

// Function to delete from the front
// of the linked list
void deleteFirst()
{
        struct node* temp;
```

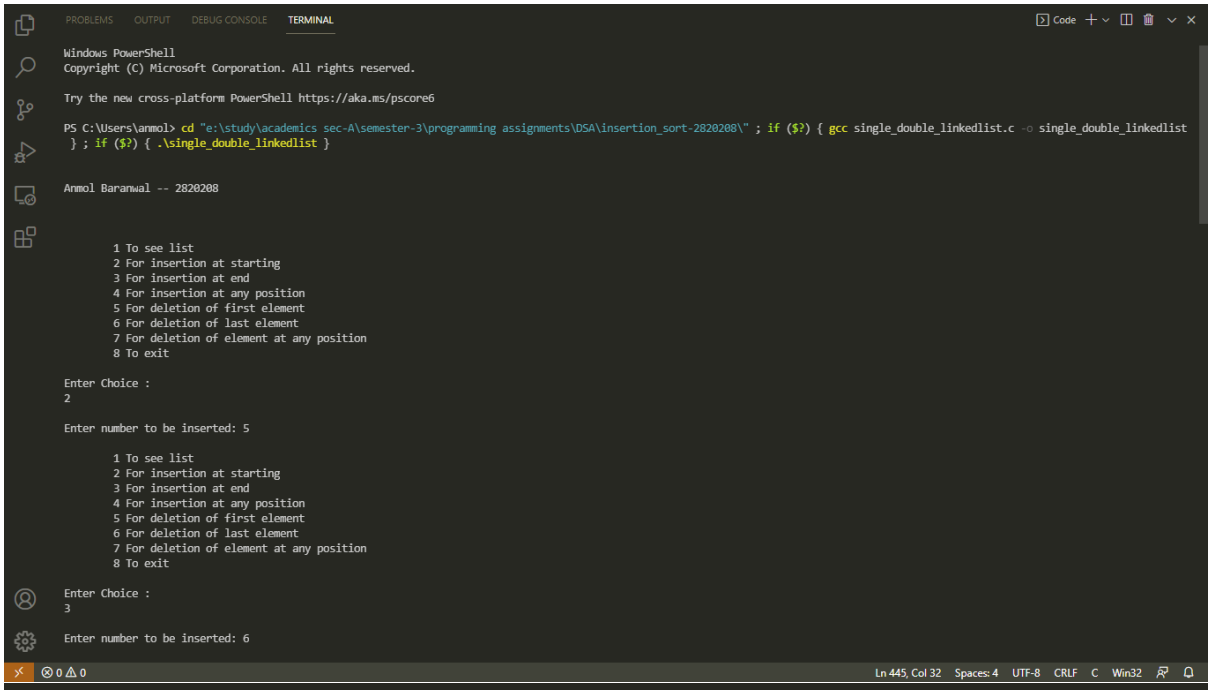
```
        if (start == NULL)
                printf("\nList is empty\n");
        else {
                temp = start;
                start = start->next;
                if (start != NULL)
                        start->prev = NULL;
                free(temp);
        }
}

// Function to delete from the end
// of the linked list
void deleteEnd()
{
        struct node* temp;
        if (start == NULL)
                printf("\nList is empty\n");
        temp = start;
        while (temp->next != NULL)
                temp = temp->next;
        if (start->next == NULL)
                start = NULL;
        else {
                temp->prev->next = NULL;
                free(temp);
        }
}

// Function to delete from any specified
```

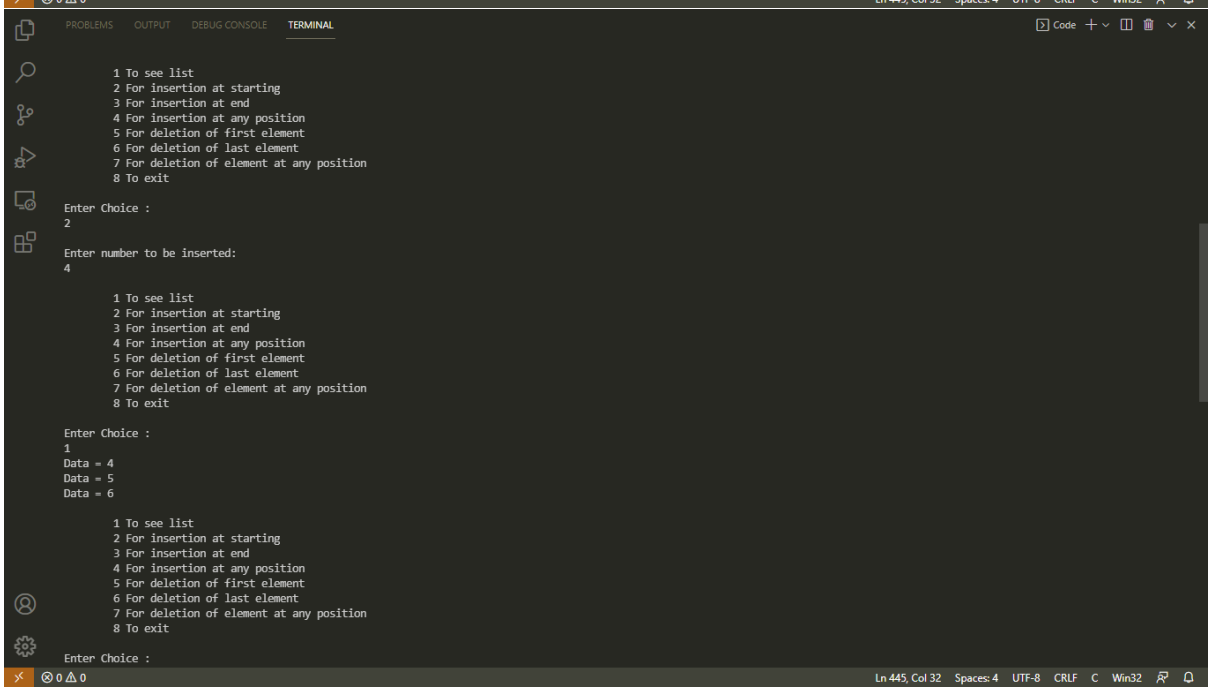
```
// position from the linked list
void deletePosition()
{
	int pos, i = 1;
	struct node *temp, *position;
	temp = start;

	// If DLL is empty
	if (start == NULL)
		printf("\nList is empty\n");

	// Otherwise
	else {
		// Position to be deleted
		printf("\nEnter position : ");
		scanf("%d", &pos);

		// If the position is the first node
		if (pos == 1) {
			deleteFirst(); // im,proved by Jay Ghughriwala on GeeksforGeeks
			if (start != NULL) {
				start->prev = NULL;
			}
			free(position);
			return;
		}

		// Traverse till position
		while (i < pos - 1) {
			temp = temp->next;
```

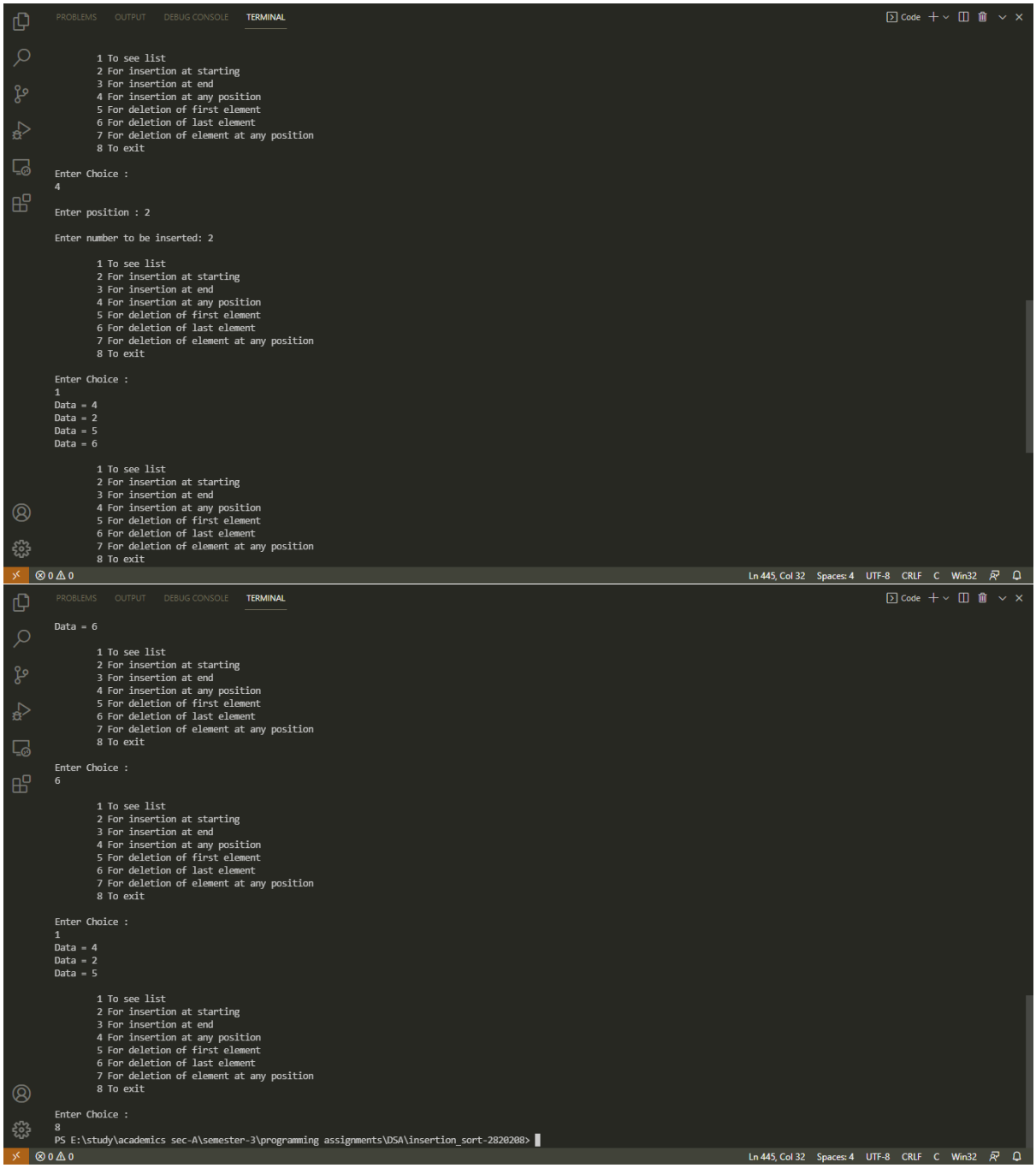
```
                i++;
            }
            // Change Links
            position = temp->next;
            if (position->next != NULL)
                position->next->prev = temp;
            temp->next = position->next;

            // Free memory
            free(position);
        }
}

// Driver Code
int main()
{
        printf("\n\nAnmol Baranwal -- 2820208 \n\n\n");
        int choice;
        while (1) {

            printf("\n\t1 To see list\n");
            printf("\t2 For insertion at"
                " starting\n");
            printf("\t3 For insertion at"
                " end\n");
            printf("\t4 For insertion at "
                "any position\n");
            printf("\t5 For deletion of "
                "first element\n");
            printf("\t6 For deletion of "
```

```
        "last element\n");
printf("\t7 For deletion of "
        "element at any position\n");
printf("\t8 To exit\n");
printf("\nEnter Choice :\n");
scanf("%d", &choice);

switch (choice) {
case 1:
        traverse();
        break;
case 2:
        insertAtFront();
        break;
case 3:
        insertAtEnd();
        break;
case 4:
        insertAtPosition();
        break;
case 5:
        deleteFirst();
        break;
case 6:
        deleteEnd();
        break;
case 7:
        deletePosition();
        break;
```

```
            case 8:
                exit(1);
                break;
            default:
                printf("Incorrect Choice. Try Again \n");
                continue;
            }
        }
        return 0;
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\anmol> cd "e:\study\academics sec-A\semester-3\programming assignments\DSA\insertion_sort-2820208\" ; if ($?) { gcc single_double_linkedlist.c -o single_double_linkedlist } ; if ($?) { .\single_double_linkedlist }
Anmol Baranwal -- 2820208
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
8 To exit
Enter Choice :
2
Enter number to be inserted: 5
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
8 To exit
Enter Choice :
3
Enter number to be inserted: 6
0 0
Ln 445, Col 32
Spaces: 4
UTF-8
CRLF
C
Win32


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
8 To exit
Enter Choice :
2
Enter number to be inserted:
4
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
8 To exit
Enter Choice :
1
Data = 4
Data = 5
Data = 6
1 To see list
2 For insertion at starting
3 For insertion at end
4 For insertion at any position
5 For deletion of first element
6 For deletion of last element
7 For deletion of element at any position
8 To exit
Enter Choice :
0 0
Ln 445, Col 32
Spaces: 4
UTF-8
CRLF
C
Win32



Question 10

Write a program to implement circular linked list for the following operations: Create, Display, inserting, counting, searching, traversing and deletion

Answer 10

```
#include <bits/stdc++.h>

using namespace std;

struct Node {
  int data;
  struct Node* next;
};

struct Node* addToEmpty(struct Node* last, int data) {
  if (last != NULL) return last;

  // allocate memory to the new node
  struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));

  // assign data to the new node
  newNode->data = data;

  // assign last to newNode
  last = newNode;

  // create link to iteself
  last->next = last;

  return last;
}

// add node to the front
struct Node* addFront(struct Node* last, int data) {
```

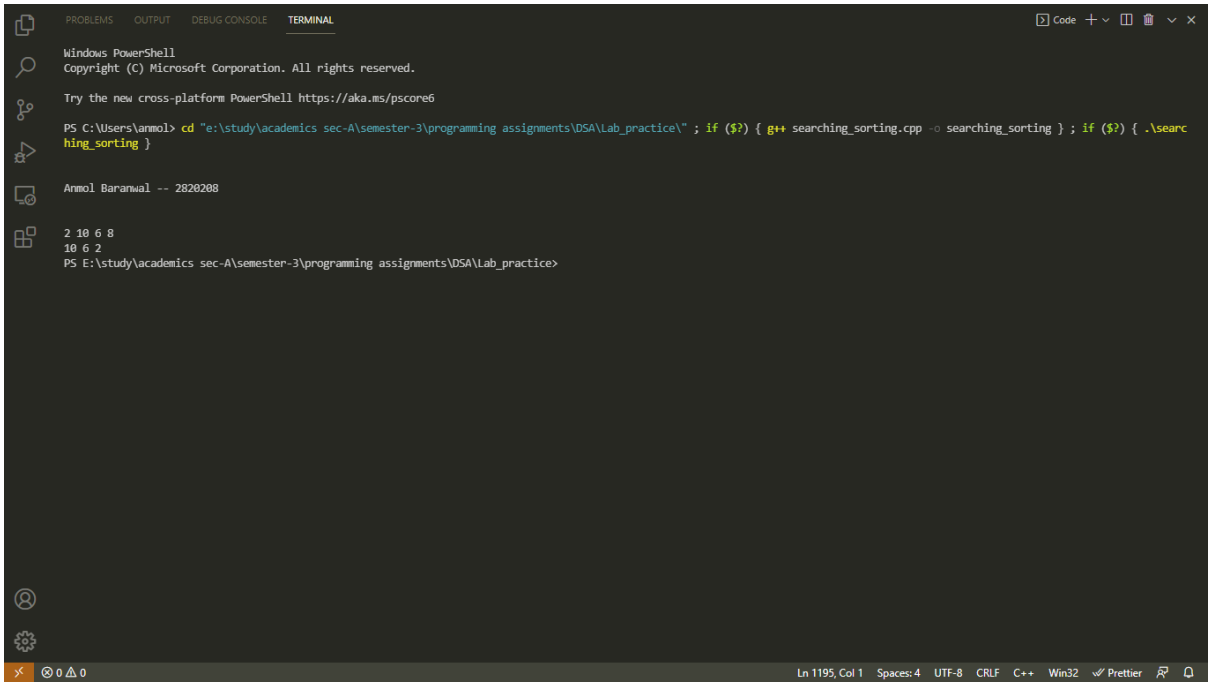
```
  // check if the list is empty
  if (last == NULL) return addToEmpty(last, data);

  // allocate memory to the new node
  struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));

  // add data to the node
  newNode->data = data;

  // store the address of the current first node in the newNode
  newNode->next = last->next;

  // make newNode as head
  last->next = newNode;

  return last;
}

// add node to the end
struct Node* addEnd(struct Node* last, int data) {
  // check if the node is empty
  if (last == NULL) return addToEmpty(last, data);

  // allocate memory to the new node
  struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));

  // add data to the node
  newNode->data = data;

  // store the address of the head node to next of newNode
```

```
  newNode->next = last->next;

  // point the current last node to the newNode
  last->next = newNode;

  // make newNode as the last node
  last = newNode;

  return last;
}

// insert node after a specific node
struct Node* addAfter(struct Node* last, int data, int item) {
  // check if the list is empty
  if (last == NULL) return NULL;

  struct Node *newNode, *p;

  p = last->next;
  do {
  // if the item is found, place newNode after it
  if (p->data == item) {
    // allocate memory to the new node
    newNode = (struct Node*)malloc(sizeof(struct Node));

    // add data to the node
    newNode->data = data;

    // make the next of the current node as the next of newNode
    newNode->next = p->next;
```

```
    // put newNode to the next of p
    p->next = newNode;

    // if p is the last node, make newNode as the last node
    if (p == last) last = newNode;
    return last;
  }

  p = p->next;
  } while (p != last->next);

  cout << "\nThe given node is not present in the list" << endl;
  return last;
}

// delete a node
void deleteNode(Node** last, int key) {
  // if linked list is empty
  if (*last == NULL) return;

  // if the list contains only a single node
  if ((*last)->data == key && (*last)->next == *last) {
  free(*last);
  *last = NULL;
  return;
  }

  Node *temp = *last, *d;
```

```
  // if last is to be deleted
  if ((*last)->data == key) {
  // find the node before the last node
  while (temp->next != *last) temp = temp->next;

  // point temp node to the next of last i.e. first node
  temp->next = (*last)->next;
  free(*last);
  *last = temp->next;
  }

  // travel to the node to be deleted
  while (temp->next != *last && temp->next->data != key) {
  temp = temp->next;
  }

  // if node to be deleted was found
  if (temp->next->data == key) {
  d = temp->next;
  temp->next = d->next;
  free(d);
  }
}

void traverse(struct Node* last) {
  struct Node* p;

  if (last == NULL) {
  cout << "The list is empty" << endl;
  return;
```

```
  }

  p = last->next;

  do {
  cout << p->data << " ";
  p = p->next;

  } while (p != last->next);
}

int main() {

  cout<<"\n\nAnmol Baranwal -- 2820208 \n\n\n";
  struct Node* last = NULL;

  last = addToEmpty(last, 6);
  last = addEnd(last, 8);
  last = addFront(last, 2);

  last = addAfter(last, 10, 2);

  traverse(last);

  deleteNode(&last, 8);
  cout << endl;

  traverse(last);
  return 0;
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\anmol> cd "e:\study\academics sec-A\semester-3\programming assignments\DSA\Lab_practice\" ; if ($?) { g++ searching_sorting.cpp -o searching_sorting } ; if ($?) { .\searching_sorting }
Anmol Baranwal -- 2820208
2 10 6 8
10 6 2
PS E:\study\academics sec-A\semester-3\programming assignments\DSA\Lab_practice>
0 0
Ln 1195, Col 1
Spaces: 4
UTF-8
CRLF
C++
Win32
Prettier