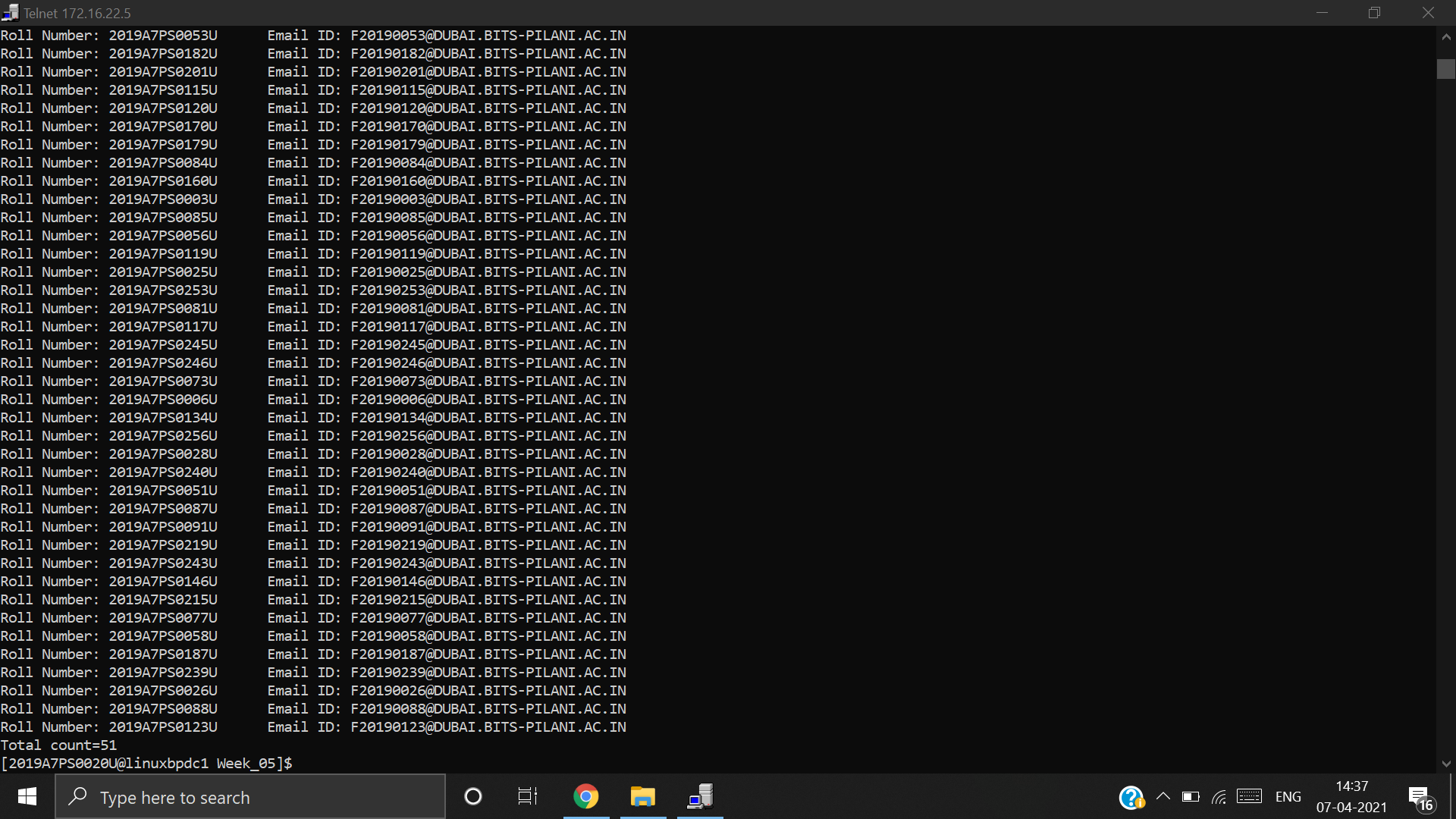
Lab 5

Ques 1:

|  |  |
| --- | --- |
| Function | Big O |
|  |  |
| struct student  {  char rollNo[20];  char emailId[35];  char lecture[3];  char tutorial[3];  char practical[3];  struct student\*next;  }; | **O(1)[declaration]** |
| struct student\*createNode(struct student buffer)  {  struct student\*ptr=(struct student\*)malloc(sizeof(struct student));  strcpy(ptr->rollNo,buffer.rollNo);  strcpy(ptr->emailId,buffer.emailId);  strcpy(ptr->lecture,buffer.lecture);  strcpy(ptr->tutorial,buffer.tutorial);  strcpy(ptr->practical,buffer.practical);  ptr->next=NULL;  return ptr;  } | O(1)[memory allocation]    O(1)[strcpy function]  O(1)[making ptr next to null]  O(1)[return stmt]  **O(1)** |
| int insertNode(struct student\*\*startPtr,struct student\*temp)  {  if(\*startPtr==NULL)  {  \*startPtr=temp;  return 1;  }  struct student\*iterator=\*startPtr;  while(iterator->next!=NULL)  {  if(strcmp(iterator->rollNo,temp->rollNo)==0)  return 0;  iterator=iterator->next;  }  if(strcmp(iterator->rollNo,temp->rollNo)==0)  return 0;  iterator->next=temp;  return 1;  } | O(1)[comparison]  O(1)[assignment]  O(1)[return]  O(1)[declaration, assignment]  O(n)[traversing the linked list till the end]  n\*O(1)[comparison, comparison]  n\*O(1)[return]  n\*O(1)[making iterator point to next iterator position]  O(1)[comparison, comparison]  O(1)[return]  O(1)[making iterator point to location of temp]  O(1)[return]  **O(n)** |
| int deleteNode(struct student\*\*ptr,char rollNumber[20])  {  struct student \*old,\*temp;  temp= \*ptr;  while(temp!=NULL)  {  if(strcmp(temp->rollNo,rollNumber)==0)  {  if(temp== \*ptr)  {  \*ptr=temp->next;  free(temp);  return 1;  }  else  {  old->next=temp->next;  free(temp);  return 1;  }  }  else  {  old=temp;  temp=temp->next;  }  }  return 0;  } | O(1)[declaration]  O(1)[assignment]  O(n)[traversing the linked list till the end]  n\*O(1)[comparison, comparison]  all the statements will take atmost n\*O(1) time for execution  **O(n)** |
| int searchNode(struct student\*ptr,char rollNumber[20])  {  struct student \*temp = ptr;  int index=1;  while(temp != NULL)  {  if(strcmp(temp->rollNo,rollNumber)==0)  {  return index;  }  else  {  temp = temp->next;  index++;  }  }  return 0;  } | O(1)[declaration, assignment]  O(1)[declaration, assignment]  O(n)[traversing the linked list till the end]  all the statements will take atmost n\*O(1) time for execution  **O(n)** |
| int displayList(struct student\*start)  {  int count=0;  struct student \*temp=start;  while(temp!=NULL)  {  printf("\nRoll Number: %s \tEmail ID: %s", temp->rollNo, temp->emailId);  temp=temp->next;  count++;  }  return count;  } | O(1)[declaration, assignment]  O(1)[declaration, assignment]  O(n)[traversing the linked list till the end]  all the statements will take atmost n\*O(1) time for execution  O(1)  **O(n)** |



Ques 2:

|  |  |
| --- | --- |
| Function | Big O |
|  |  |
| int push(struct student\*\*top,struct student\* temp) {  if(\*top==NULL) {  \*top = temp;  return 1;  }  temp->next = \*top;  \*top = temp;  } | O(1)[comparison]  O(1)[assignment]  O(1)[return stmt]  O(1)[accessing location, assignment to temp]  O(1)[assigning location of temp to \*temp]  **O(1)** |
| struct student\* pop(struct student\*\*top) {  if(\*top == NULL) {  return NULL;  }  struct student\* temp = \*top;  \*top = temp->next;  return temp;  }; | O(1)[comparison]  O(1)[return stmt]  O(1)[declaration, assignment]  O(1)[assigning location of temp->next to \*linked list, return stmt]  **O(1)** |

