Lab 6

|  |  |
| --- | --- |
| Function | Big O |
|  |  |
| int \*queue;  int size;  int \*stack1,\*stack2;  int front=0;int rear=0;  struct node\* head1;  struct node\* head2; | **O(1)[declaration and assignment]** |
| struct node  {  int data;  struct node \*next;  }; | **O(1)[declaration]** |
| struct node\* insert(int x)  {  struct node\*temp = (struct node\*)malloc(sizeof(struct node));  temp->data = x;  temp->next = NULL;  return temp;  } | O(1)[memory allocation]  O(1)[assigning to data]  O(1)[assigning null value to temp->next]  O(1)[return stmt]  **O(1)** |
| int push(int x,struct node \*\*root)  {  if(\*root==NULL)  {  struct node\* temp = insert(x);    \*root = temp;  }  struct node\* temp = insert(x);    temp->next = \*root;  \*root = temp;  return 1;  } | O(1)[comparison]  O(1)[creation and assigning values from insert function]  O(1)[assigning location of temp to root]  O(1)[creation and assigning values from insert function]  O(1)[adding new node at beginning and swapping location of root and temp]  **O(1)** |
| int pop(struct node\*\*root)  {  if(\*root == NULL)  {  return 0;  }  struct node\* temp = \*root; //check wheter \*root or root  int temp\_data = temp->data;  \*root = temp->next;  return temp\_data;  } |  |
| int enqueue(int x)  {  push(x,&head2);  printf("Enqueued %d\n",x);  rear++;  } | O(1)[function call]  O(1)[print stmt]  O(1)[increment]  **O(1)** |
| int dequeue()  {  if(front==rear)  return 0;  int x=0;  struct node\* temp2 = head2;  while(temp2!=NULL)  {  x = pop(&head2);  push(x,&head1);  temp2 = temp2->next;  }    int y = pop(&head1);  struct node\* temp1 = head1;  while(temp1!=NULL)  {  x = pop(&head1);  push(x,&head2);  temp1 = temp1->next;  }    front++;  return y;  } | O(1)[comparison]  O(1)[return stmt]  O(1)[declaration, assignment]  O(1)[declaration, assignment]  O(n)[traversing the linked list till the end]  n\*O(1)[assignment, function call to pop()]  n\*O(1)[function call to push()]  n\*O(1)[making pointer to move to next element in linked list]  O(1)[function call to pop(), assignment]  O(1)[declaration, assignment]  O(n)[traversing the linked list till the end]  n\*O(1)[assignment, function call to pop()]  n\*O(1)[function call to push()]  n\*O(1)[making pointer to move to next element in linked list]  O(1)[increment]  O(1)[return stmt]  **O(n)** |

