2.
```
#include <stdio.h>
# include< stdlib.h>
# include < string.h>

struct staff {
        int id;
        char name [50];
        char dept [20];
        struct staff *next;
   };
   struct staff * head = NULL;
void addAtBeginning (int id, char name[],
                     char dept[])
   {
     struct staff *newnode = (struct staff *)
                     malloc (sizeof(struct
                                    staff));
       new node -> id = id;
       strcpy (newnode -> name, name);
       strcpy (newnode -> dept, dept);
       newnode -> next = head;
         head = newnode;

   }
void addAtEnd (int id, char name[],
                 char dept []) {

    struct staff *newnode = (struct staff*)
                     malloc (sizeof (
                             struct staff))

      new node -> id = id;
    strcpy (newnode -> name, name);
strcpy (newnode -> dept , dept);
newnode -> next = NULL;
if (head == NULL){
        head = newnode;
        return;
    }
  struct staff *temp = head;
  while (temp -> next != NULL){
        temp = temp -> next;
    }
   temp -> next = newnode;

}
void insertAtPosition (int pos, int id, char name[],
                       char dept[])
{
  struct staff *newnode = (struct staff*) malloc
                      (sizeof (struct staff));

   newnode -> id = id;
   strcpy (newnode -> name, name)
   strcpy (newnode -> dept, dept)
   if (pos == 1) {
      newnode -> next = head;
       head = newnode;
      return;
   }
  struct staff *temp = head
  for (int i = 1 ; temp != Null && i<pos-1 ; i++)
       temp = temp -> next;
```

```
if (temp==NULL) {
    printf ("position out of range ! \n");
    free (newnode);
    return;
}
newnode ->next = temp->next;
temp-> next= newnode;
}
void delete By Position (int pos){
        if (head==NULL){
            printf("list is empty\n")
            return;
        }
    struct staff * temp= head;
    if (pos==1){
        head = temp ->next;
        free (temp);
        return;

    }
    struct staff *prev= NULL;
    for (int i=1 ; temp!=NULL && i<pos ;i++)
    {
        prev= temp;
        temp = temp->next;

    }
    if (temp ==NULL){
        printf (" Position out of range \n");
        return;

    }
```

```
prev -> next = temp -> next;
 free(temp);

 y
void searchstaff(){
      int choice, id;
      char name [50];
      printf("1. search by staff ID \n
             2. search by Name \n
                 Enter choice: ");

      scanf("%d", choice);
      struct staff * temp = head;
      if (choice == 1){
          printf ("Enter staff ID: ");
          scanf ("%d", &id);
          while (temp != NULL){
              if (temp -> id == id){
              printf("\staff found: ID
                          = %d", Name
                  = %s, dept = %s \n",
          temp->id, temp->name, temp->dept);

          return;

     y
     temp = temp -> next;

    }
    printf("staff not found \n");
 }
```

```
else if (choice==2){
        printf("Enter staff name:");
        scanf(" %s", &name);
        while (temp!=NULL){
          if (strcmp(temp->name, name)==0){
            printf("staff found: ID =%d,
                  name=%s, dept=%s \n",
           temp->id, temp->name, temp->dept);
            return;
          }
          temp= temp->next;
        }
        printf("staff not found \n");
    }
    else {
        printf( "Invalid choice \n");

    }
}
void displayList(){
        if (head==NULL){
            printf("No staff in the
                    list \n");

            return;

        }
    struct staff *temp= head;
    printf("\n--- staff Allotment List--- \n");
```

```
while (temp != NULL){
    printf("ID= %d |Name: %s | Department
                              : %s /n",
        temp->id, temp->name, temp->dept);
    temp= temp ->next;
  }
}
int main(){
  int choice, id ,pos;
  char name [50], dept [50];
  while(1){
  printf("\n--- Staff Allotment menu---\n");
   printf(" 1. add staff at Beginning \n");
   printf("2. Add staff at End \n");
   printf("3. Insert at position \n");
   printf("4. Delete by position \n");
   printf(" 5. search staff \n");
   printf(" 6. Display staff list \n");
   printf(" 7. Exit \n");
   while(1){
     printf("Enter your choice: ");
      scanf(" %d", &choice);
     switch (choice){
      case1: printf("Enter ID, Name, Department:");
         scanf("%d %s %s", &id, name, dept);
    addAtBegginning (id, name, dept);
    break;
case2: printf(" Enter ID, Name, Department:");
       scanf(" %d %s %s", &id, name, dept);
       addAtEnd (id, name, dept);
       break;

case3: printf(" Enter position :");
        scanf(" %d", &pos);
        printf(" Enter ID, Name, Department:");
        scanf(" %d %s %s", &id, name, dept);
        insertAtPosition (pos, id, name, dept);
        break;

case 4: printf (" Enter position to delete :");
         scanf(" %d", &pos);
         delete By Position (pos);

         break;
case5: search staff();
         break;

case6: display List();
         break;

case7: exit (0);
default : printf (" Invalid choice\n");
4}}
```

output:

```
--- staff Allotment menu ---
1. Add staff at beginning
2. Add staff at end
3. Insert at position
4. Delete at position
5. search staff
6. Display staff list
7. Exit
Enter choice: 1
Enter ID, name, Department :101, Rahul CSE
Enter choice : 2
Enter ID, name, Department : 102 Meera, ECE

Enter choice; 2
 Enter ID, name, Department; 103 David ME

Enter choice : 3
- Enter position: 2
Enter ID, name, Department: 104 Julie civil

Enter choice : 6
ID : 101 | name : Rahul | Department : CSE
ID : 104 | name : Julie | Department : civil
ID : 102 | name : Meera | Department : ECE
ID : 103 | name : David | Department : ME

Enter choice : 5
1. search by staff ID
2. search by name
  Enter your choice : 1
  Enter staff ID : 104
```

```
Staff Found : ID-104 , Name = Julie, Departme

Enter choice : 4
 Enter position to Delete : 3

Enter choice : 6
ID : 101 | Name : Rahul | Department : CSE
ID : 104 | Name : Julie | Department : Civil
ID : 103 | Name : David | Department : ME

Enter choice : 7
```