LAB-4

Krishna Kumar Agrawalla
190536

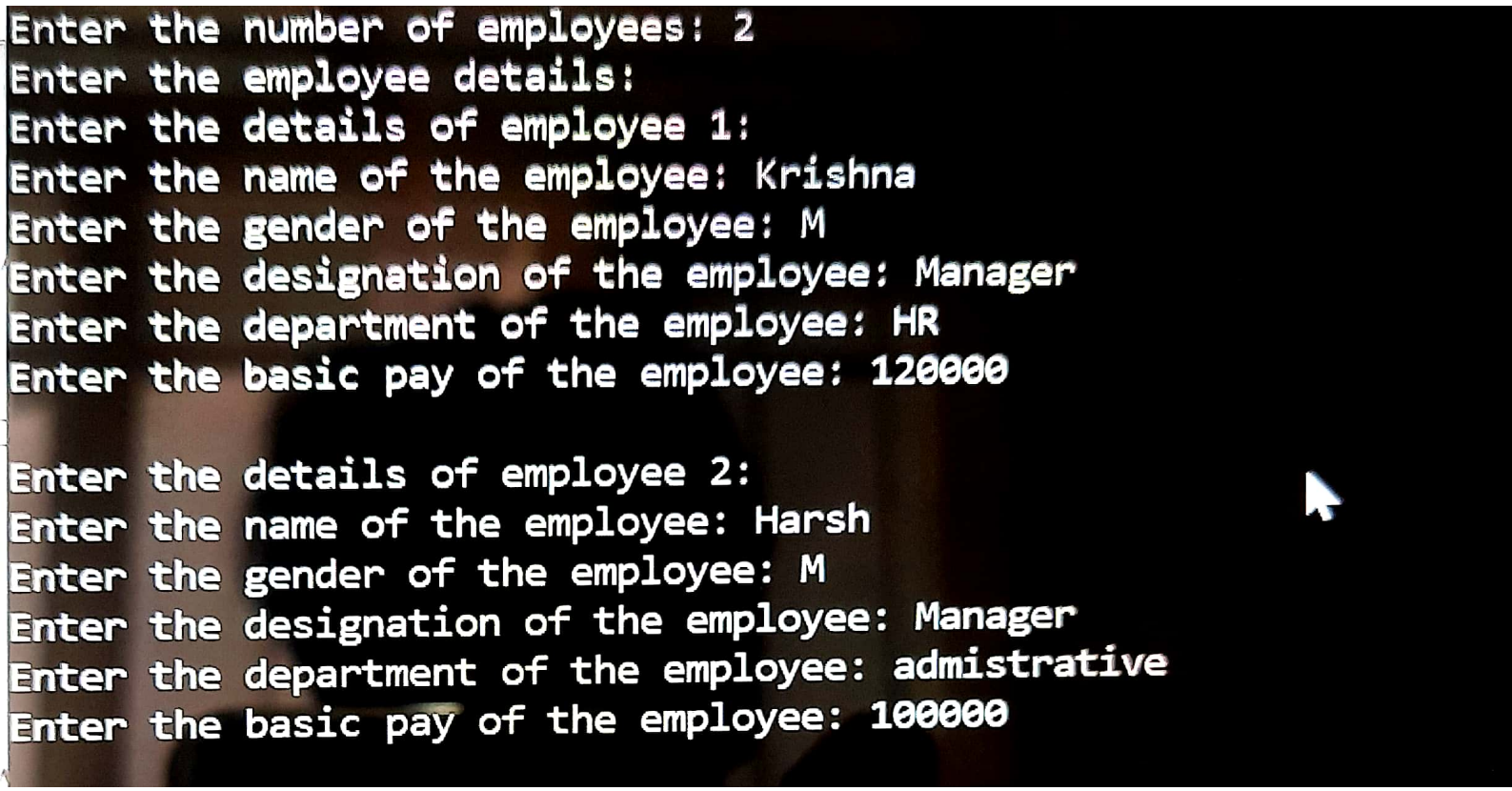
Q7: WAP to store n employees data such as employee name, gender, designation, department, basic pay. Calculate the gross pay of each employee.

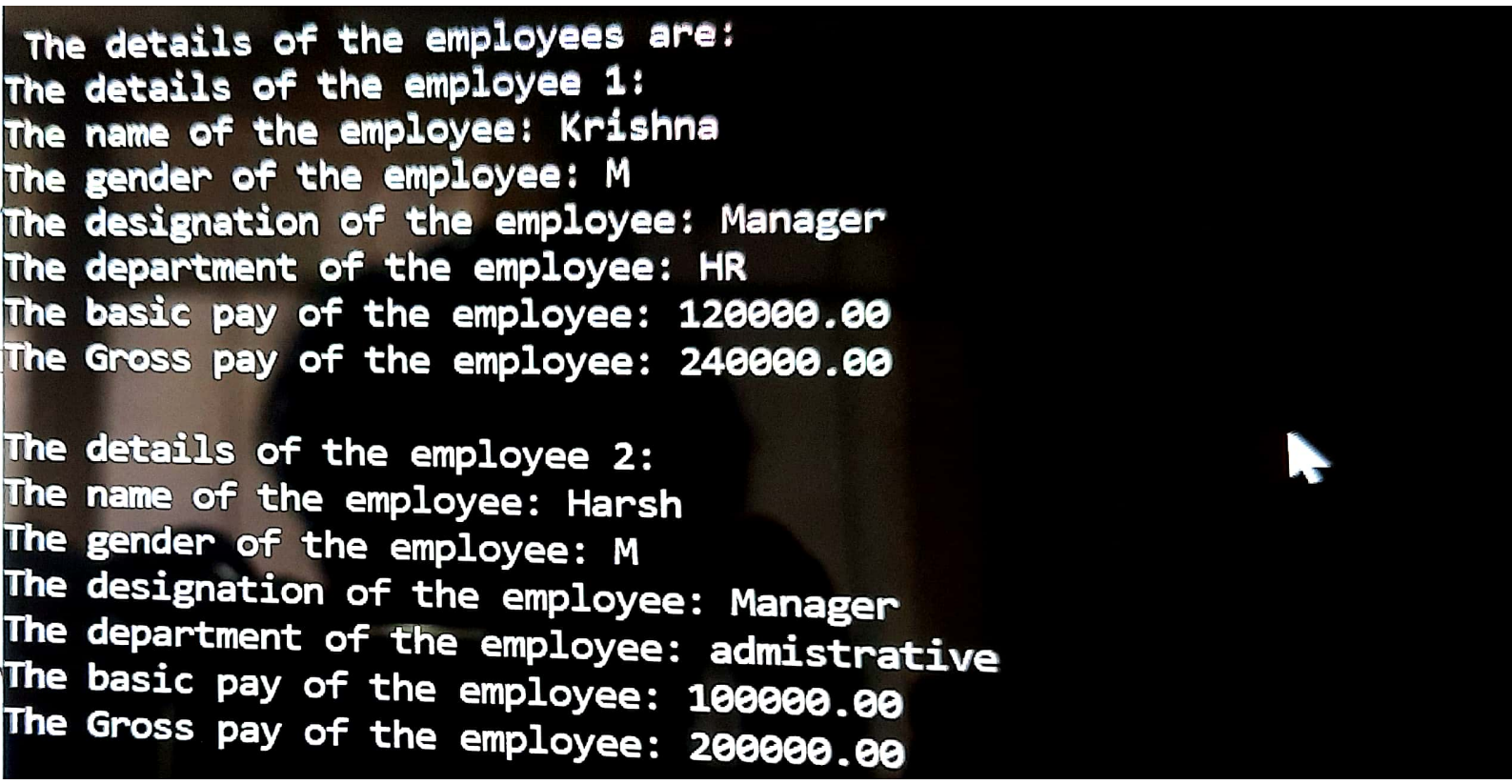
Gross pay = basic pay + HR + DA

HR = 25% of basic DA = 75% of basic

Sol:
```
#include<stdio.h>

struct employee //structure to hold different data.
{ char name[50], gen, designation[20], department[10];
  float basic, grosspay;
};

float Calculate(struct employee E) //calculating the GP
{ float gp, hr, da;
  hr = 0.25*E.basic;
  da = 0.75*E.basic;
  gp = hr+da+E.basic;
  return gp;
}

int main()
{ struct employee *e;
  int n, i;
  printf("Enter the number of employees: ");
  scanf("%d", &n);
  e = (struct employee*)malloc(n*sizeof(struct employee));
                                  //Memory allocated
  printf("Enter the employee details: ");
```

```
for(i=0;i<n;i++)          // getting input.
{ //printf("hi
  printf("\nEnter the details of Employee %d\n", i+1);
  printf("Enter the name of the employee: ");
  scanf("%s", e[i].name);
  printf("Enter the gender of the employee: ");
  scanf("%c", &e[i].gen);
  printf("Enter the designation of the employee: ");
  scanf("%s", e[i].designation);
  printf("Enter the department of the employee: ");
  scanf("%s", e[i].department);
  printf("Enter the basic pay of the employee: ");
  scanf("%f", &e[i].basic);
  e[i].grosspay = calculate(e[i]);
}

printf("\nThe details of the employees are: ");
for(i=0;i<n;i++)          // printing details
{ printf("\nThe details of employee %d\n", i+1);
  printf("The name of the employee: %s\n", e[i].name);
  printf("The gender of the employee: %c\n", e[i].gen);
  printf("The designation of the employee: %s\n", e[i].designation);
  printf("The department of the employee: %s\n", e[i].department);
  printf("The basic pay of the employee: %.2f\n", e[i].basic);
  printf("The gross pay of the employee: %.2f\n", e[i].grosspay);
}

return 0;
}
```

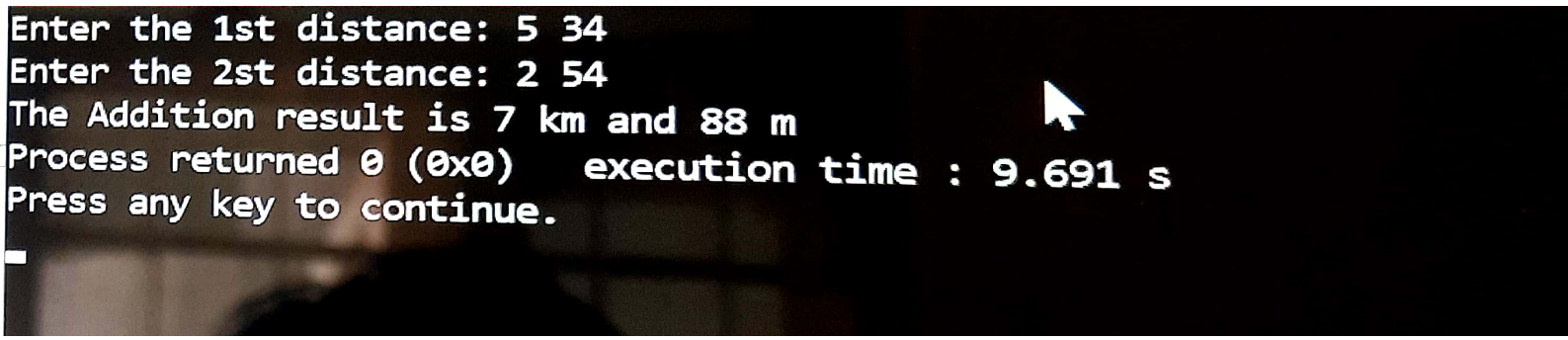

Enter the number of employees: 2
Enter the employee details:
Enter the details of employee 1:
Enter the name of the employee: Krishna
Enter the gender of the employee: M
Enter the designation of the employee: Manager
Enter the department of the employee: HR
Enter the basic pay of the employee: 120000
Enter the details of employee 2:
Enter the name of the employee: Harsh
Enter the gender of the employee: M
Enter the designation of the employee: Manager
Enter the department of the employee: admistrative
Enter the basic pay of the employee: 100000


The details of the employees are:
The details of the employee 1:
The name of the employee: Krishna
The gender of the employee: M
The designation of the employee: Manager
The department of the employee: HR
The basic pay of the employee: 120000.00
The Gross pay of the employee: 240000.00
The details of the employee 2:
The name of the employee: Harsh
The gender of the employee: M
The designation of the employee: Manager
The department of the employee: admistrative
The basic pay of the employee: 100000.00
The Gross pay of the employee: 200000.00

Q8. WAP to add 2 distances (in km-meter) by passing structure to a function

Sol^n:

```
#include <stdio.h>
struct distance
{  int km, m;
};

void add (struct distance d1, struct distance d2)
{ struct distance d3;          // structure variable
  d3.km = d1.km + d2.km;
  d3.m = d1.m + d2.m;
  if (d3.m >= 1000)           // condition for conversion
  { d3.km += 1;
    d3.m -= 1000;
  }
  printf("The Addition result is %d km and %d m",
                    d3.km, d3.m);
}

void main()
{ struct distance d1, d2;     // structure variable
  printf("Enter the 1st distance : ");
  scanf("%d %d", &d1.km, &d1.m);

  printf("Enter the 2nd distance : ");
  scanf("%d %d", &d2.km, &d2.m);

  add(d1, d2);                // function call.
  return 0;
}
```


Enter the 1st distance: 5 34
Enter the 2st distance: 2 54
The Addition result is 7 km and 88 m
Process returned 0 (0x0)   execution time : 9.691 s
Press any key to continue.

Q3. Write a menu driven program to perform the following operations in a single linked list by using suitable user defined functions for each case

a) Transversal of the list.
b) check if the list is empty.
c) Insert a node at the certain position
d) Delete a node at the certain position
e) Delete a node for the given key.
f) Count the total no. of nodes.
g) Search for an element in the linked list.

Verify & validate each function from main method.

Sol^n:

```
#include<stdio.h>
#include<stdlib.h>
struct node
{ int data;
  struct node *next;
};
struct node *start = NULL;
void create ()
{ struct node *temp ,*ptr;
  temp = (struct node *) malloc(sizeof(struct node));
  if (temp == NULL)
  { printf("Out of Memory Space:\n");
    exit(0);
  }
  printf(" Enter the data value for the node: ");
  scanf("%d", &temp -> data);
  temp ->next = NULL;
  if (start == NULL)
      start = temp;
  else
  { ptr = start;
    while (ptr ->next!= NULL)
        ptr = ptr ->next;
    ptr ->next=temp;
  }
}
```

```
void check()
{ if(start == NULL)
      printf("The list is empty\n");
  else
      printf("The list contains nodes(s)");
}

void display()
{ struct node *ptr;
  if(start == NULL)
  {  printf("List is empty:");
  else
  { ptr = start;
    printf("The List elements are:\n");
    while(ptr != NULL)
    { printf("%d\t", ptr->data);
      ptr = ptr->next;
    }
  }
}

void insert()
{ int pos;
  struct node *ptr, *ptr1, *temp;
  temp = (struct node*) malloc(sizeof(struct node));
  printf("Enter the position you want to insert:");
  scanf("%d", &pos);
  printf("Enter the data value of the node:");
  scanf("%d", temp->data);
  ptr = start;
  for(int i=1; i<pos; i++)
  { if(ptr->next == NULL)
        break;
    ptr1 = ptr;
    ptr = ptr->next;
  }
  ptr1->next = temp;
  temp->next = ptr;
  printf("The list after insertion:\n");
  display();
}
```

```
void make()
{ int cont;
  printf("Do you want to enter the value to node? (0/1) ");
  Scanf("%d",&cont);
  while (cont==1)
  {  create();
     printf("\nDo you want to enter more node?(0/1)");
     scanf("%d",&cont);
  }
}

void deletion()
{  int pos;
   struct node *ptr, *ptr1;
   printf("Enter the position you want to delete: ");
   scanf("%d",&pos);
   ptr=start;
   for(int i=1; i<pos; i++);
   {   if(ptr->next==NULL)
          break;
       ptr1=ptr;
       ptr=ptr->next;
   }
   ptr1->next=ptr->next;
   printf("The next list after deletion is:\n");
   display();
}

void deletekey()
{  int key;
   struct node *ptr, *ptr1;
   printf("Enter the key you want to delete: ");
   scanf("%d",&key);
   ptr=start;
   while(ptr->data!=key)
   {  if(ptr->next==NULL)
      {  printf("key not found");
         break;
      }
      ptr1=ptr;
      ptr=ptr->next;
   }
```

```
    ptr1->next=ptr->next;
    printf("The new list after deletion is:\n");
    display();
}

void countnodes()
{   int count=0;
    struct node *ptr;
    ptr=start;
    while(ptr!=NULL)
    {   count++;
        ptr=ptr->next;
    }
    printf("The number of nodes in the list are: %d", count);
}

void search()
{   struct node *ptr;
    int pos=0,item;
    printf("Enter the element you want to search=");
    scanf("%d",&item);
    ptr=start;
    while(ptr!=NULL)
    {   pos++;
        if(ptr->data==item)
        {   printf("The element was found in %d position", pos);
            break;
        }
        ptr=ptr->next;
    }
    if(ptr==NULL)
        printf("The element was not found");
}
```

```
int main()
{ char ch;
  do
  { int choice;
    printf("\n**** MENU*****");
    printf("\n1. Make a list");
    printf("\n2. Transversal of the list");
    printf("\n3. Check if the list is empty.");
    printf("\n4. Insert a node at the certain position");
    printf("\n5. Delete a node at a certain Position");
    printf("\n6. Delet a node for the given key");
    printf("\n7. Count the total number of nodes.");
    printf("\n8. Search for an element in the list");
    printf("\n Enter your choice: ");
    scanf("%d", &choice);
    switch(choice)
    { case 1: make()
              break;
      case 2: display()
              break;
      case 3: check();
              break;
      case 4: insert();
              break;
      case 5: deletion();
              break;
      case6: deletekey();
              break;
      case7: countnodes();
              break;
      case 8: search()
              break;
    }
    printf("Doyou want to continue (y/n): ");
    scanf("%c", &ch);
  } while(ch == 'y');
  return 0;
}
```

```
********** MENU **********
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 1
Do you want to enter the value to node?(0/1) 1
Enter the data value for the node: 45

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 46

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 48

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 49

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 50

Do you want to enter more node?(0/1) 0

Do you want to continue(y/n): y
```

```
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 2
The List elements are:
45      46      48      49      50
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 3
The list contains node(s)
Do you want to continue(y/n): y
```

```
********** MENU **********
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 4
Enter the position you want to insert: 3
Enter the data value of the node: 47
The next list after insertion is:
The List elements are:
45      46      47      48      49      50
Do you want to continue(y/n): y
```

```
The List elements are:
45      46      47      48      49      50
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 5
Enter the position you want to delete: 6
The new list after deletion is:
The List elements are:
45      46      47      48      49
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 6
```

```
"C:\COdeBLock\DSA LAB\LAB 4\Link List menu driven\bin\Debu
The List elements are:
45      46      47      48      49
Do you want to continue(y/n): y

********** MENU **********
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 6
Enter the key you want to delete: 47
The new list after deletion is:
The List elements are:
45      46      48      49
Do you want to continue(y/n): y

********** MENU **********
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 7
```

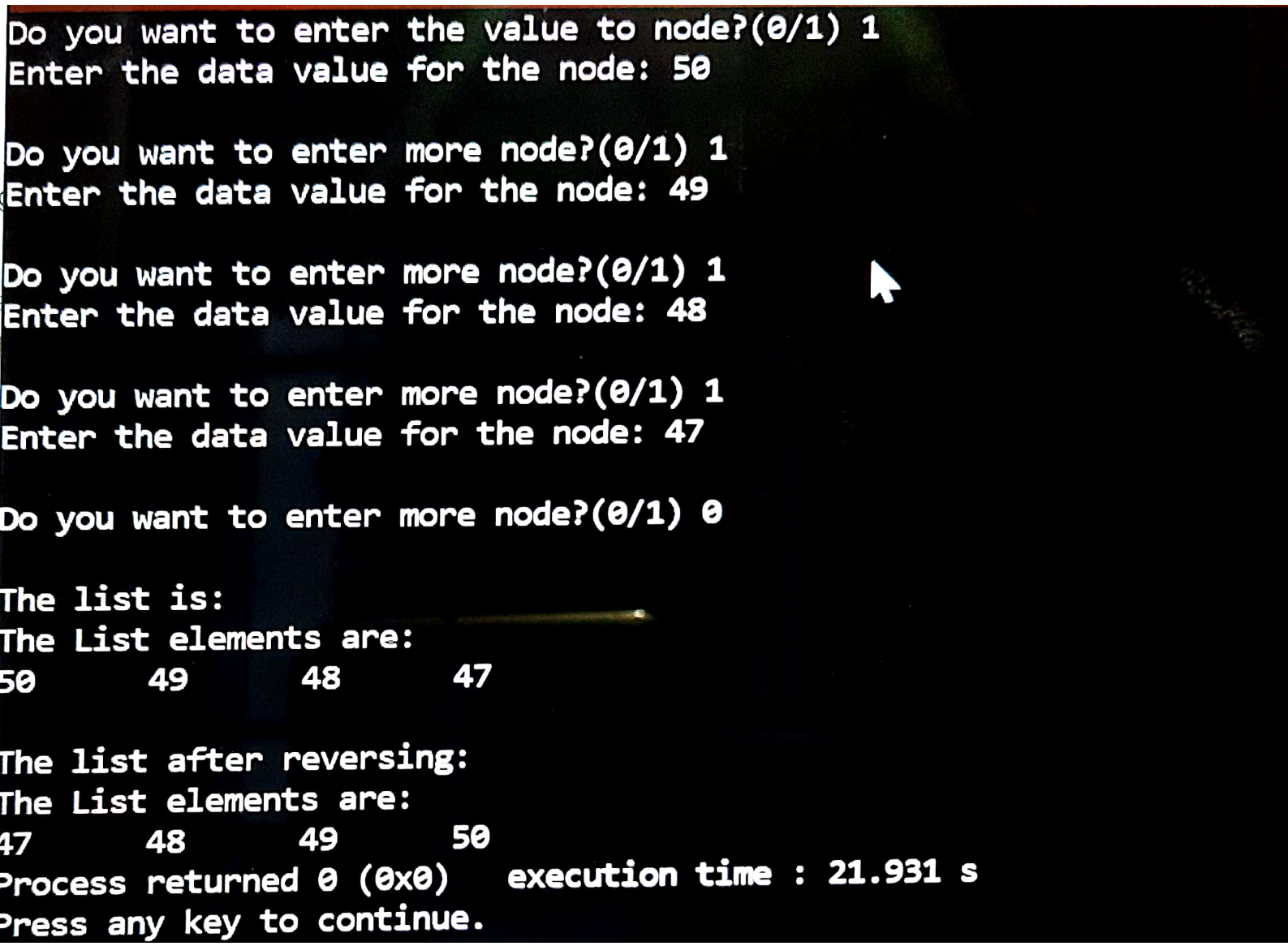
```
The List elements are:
45      46      48      49
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 7
The number of nodes in the list are: 4
Do you want to continue(y/n): y

************ MENU ************
1. Make a list
2. Traversal of the list
3. Check if the list is empty.
4. Insert a node at the certain position
5. Delete a node at the certain position
6. Delete a node for the given key
7. Count the total number of nodes
8. Search for an element in the linked list
Enter your choice: 8
Enter the element you want to search: 48
The element was found in 3 position
Do you want to continue(y/n):
```

Q4. WAP to display the contents of a linked list in reverse order.

Sol^n:

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

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

struct node *start = NULL;

void create()
{ struct node *temp, *ptr;
  temp = (struct node *) malloc (sizeof(struct node));
  if(temp == NULL)
  { printf("Out of memory space\n");
    exit(0);
  }
  printf("Enter the data value for the node: ");
  scanf("%d", &temp->data);
  temp->next = NULL;
  if(start == NULL)
     start = temp;
  else
  { ptr = start;
    while(ptr->next != NULL)
       ptr = ptr->next;
    ptr->next = temp;
  }
}
```

```
void display()
{  struct node *ptr;
    if(start == NULL)
    {  printf(" List is empty ");
    else
    {  ptr =start;
       printf(" The list elements are: \n ");
       while (ptr!= NULL)
       { printf("%d \t", ptr->data);
         ptr = ptr->next;
       }
   }
}

void reverse()
{  struct node * curr =NULL, *prev= NULL , *next = NULL;
    curr = start;
    while (curr! = NULL)
    {  next = curr->next;
       curr->next = prev;
       prev = curr;
        curr = next;
     }
     start = prev;
}

int main()
{  int cont;
    printf(" Do you want to enter node?(0/1)");
    scanf("%d", &cont);
    { create();
      printf(" Do you want to enter more?(0/1)");
       scanf("%d", &cont);
    }
    printf(" The list is:\n");
    display();
    reverse();
    printf(" \n\n The list after reversing : \n ");
    display();
    return 0;
}
```


Do you want to enter the value to node?(0/1) 1
Enter the data value for the node: 50
Do you want to enter more node?(0/1) 1
Enter the data value for the node: 49
Do you want to enter more node?(0/1) 1
Enter the data value for the node: 48
Do you want to enter more node?(0/1) 1
Enter the data value for the node: 47
Do you want to enter more node?(0/1) 0
The list is:
The List elements are:
50 49 48 47
The list after reversing:
The List elements are:
47 48 49 50
Process returned 0 (0x0) execution time : 21.931 s
Press any key to continue.

Q5 WAP to print mth node from last of the linked list of n nodes.

Sol^n:

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

struct node
{  int data;
   struct node *next;
};
struct node *start = NULL;

void create()
{  struct node *temp, *ptr;
   temp = (struct node *) malloc (sizeof(struct node));
   if(temp == NULL)
   {  printf("Out of Memory Space: \n");
      exit(0);
   }
   printf("Enter the data value for the node: ");
   scanf("%d", &temp->data);
   temp->next = NULL;
   if(start == NULL)
        start = temp;
   else
   {  ptr = start;
      while(ptr->next != NULL)
           ptr = ptr->next;
      ptr->next = temp;
   }
}
```

```
void display()
{   struct node *ptr;
    if(start == NULL)
    {  printf("The list is empty. ");
    else
    {  ptr=start;
       printf("The list elements are :\n");
       while(ptr != NULL)
       {   printf("%d\t", ptr->data);
           ptr=ptr->next;
       }
    }
}

void mthelement()
{  struct node *ptr;
   int pos, n=0;
   printf("\nEnter the position you want to see: ");
   scanf("%d", &pos);
   ptr=start;
   while(ptr != NULL)
   {  n++;
      ptr=ptr->next;
   }
   ptr=start;
   if(pos<=n)
   {  for(int i=1; i<(n-pos+1); i++)
          ptr=ptr->next;
   printf("The element at %d position from last is
          %d\n", pos, ptr->data);
   }
   else
       printf("Position not available\n");
}
```

```
int main()
{ int cont;
  printf("Do you want to enter the value to node?(0/1) ");
  scanf("%d",&cont);
  while(cont==1)
  { create();
    printf("\nDo you want to enter more node?(0/1)");
    scanf("%d",&cont);
  }
  printf("\n The list is:\n");
  display();
  mth element();
  return 0;
}
```

```
Do you want to enter the value to node?(0/1) 1
Enter the data value for the node: 46

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 47

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 48

Do you want to enter more node?(0/1) 1
Enter the data value for the node: 49

Do you want to enter more node?(0/1) 0

The list is:
The List elements are:
46      47      48      49
Enter the positon you want to see: 2
The element at 2 position from last is 48
```