LAB PROGRAM-5

5) WAP to Implement Singly Linked List with following operations a) Create a linked list. b) Insertion of a node at first position, at any position and at end of list. c) Display the contents of the linked list.

CODE:

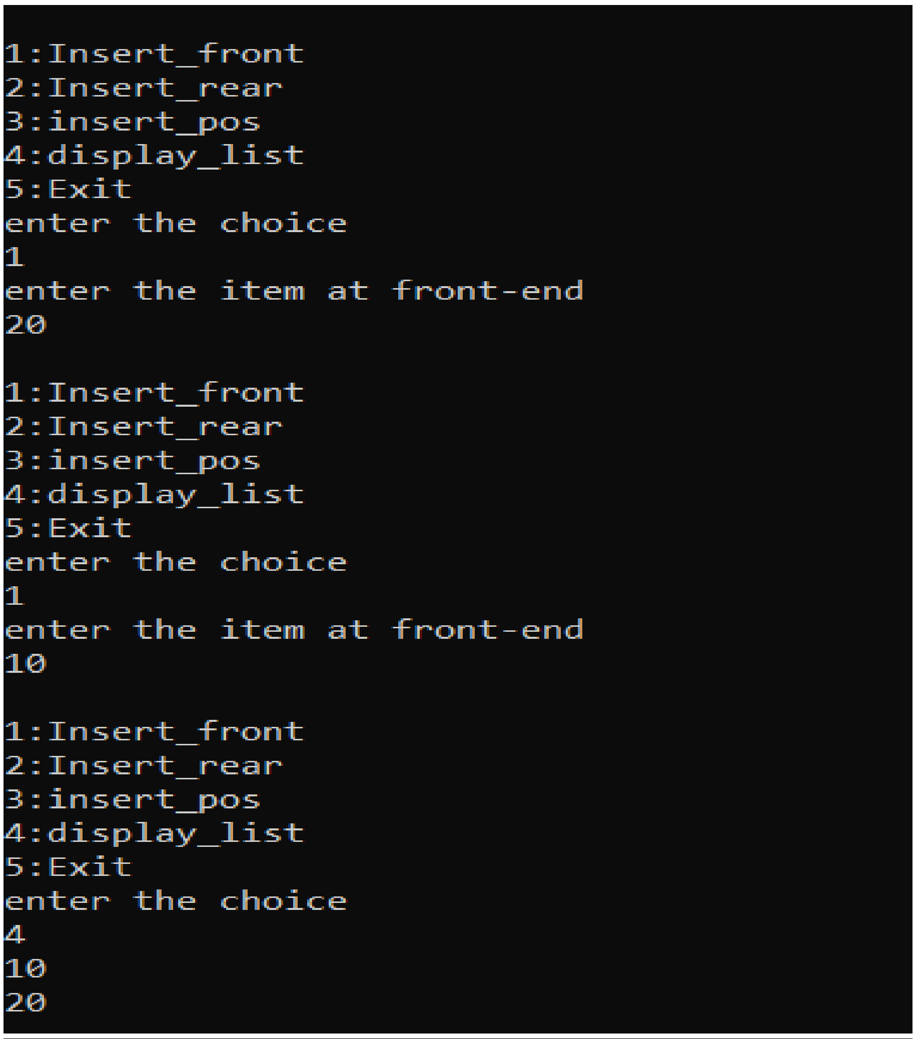
```
#include <stdio.h>
#include <conio.h>
struct node
{
int info;
struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x = (NODE)malloc(sizeof(struct node));
if (x == NULL)
{
printf("mem full\n");
exit(0);
}
return x;
}
void freenode(NODE x)
{
free(x);
}
```

```
NODE insert_front(NODE first, int item)
{
NODE temp;
temp = getnode();
temp->info = item;
temp->link = NULL;
if (first == NULL)
return temp;
temp->link = first;
first = temp;
return first;
}
NODE insert_rear(NODE first, int item)
{
NODE temp, cur;
temp = getnode();
temp->info = item;
temp->link = NULL;
if (first == NULL)
return temp;
cur = first;
while (cur->link != NULL)
cur = cur->link;
cur->link = temp;
return first;
}
NODE insert_pos(int item, int pos, NODE first)
{
NODE temp;
NODE prev, cur;
int count;
temp = getnode();
temp->info = item;
```

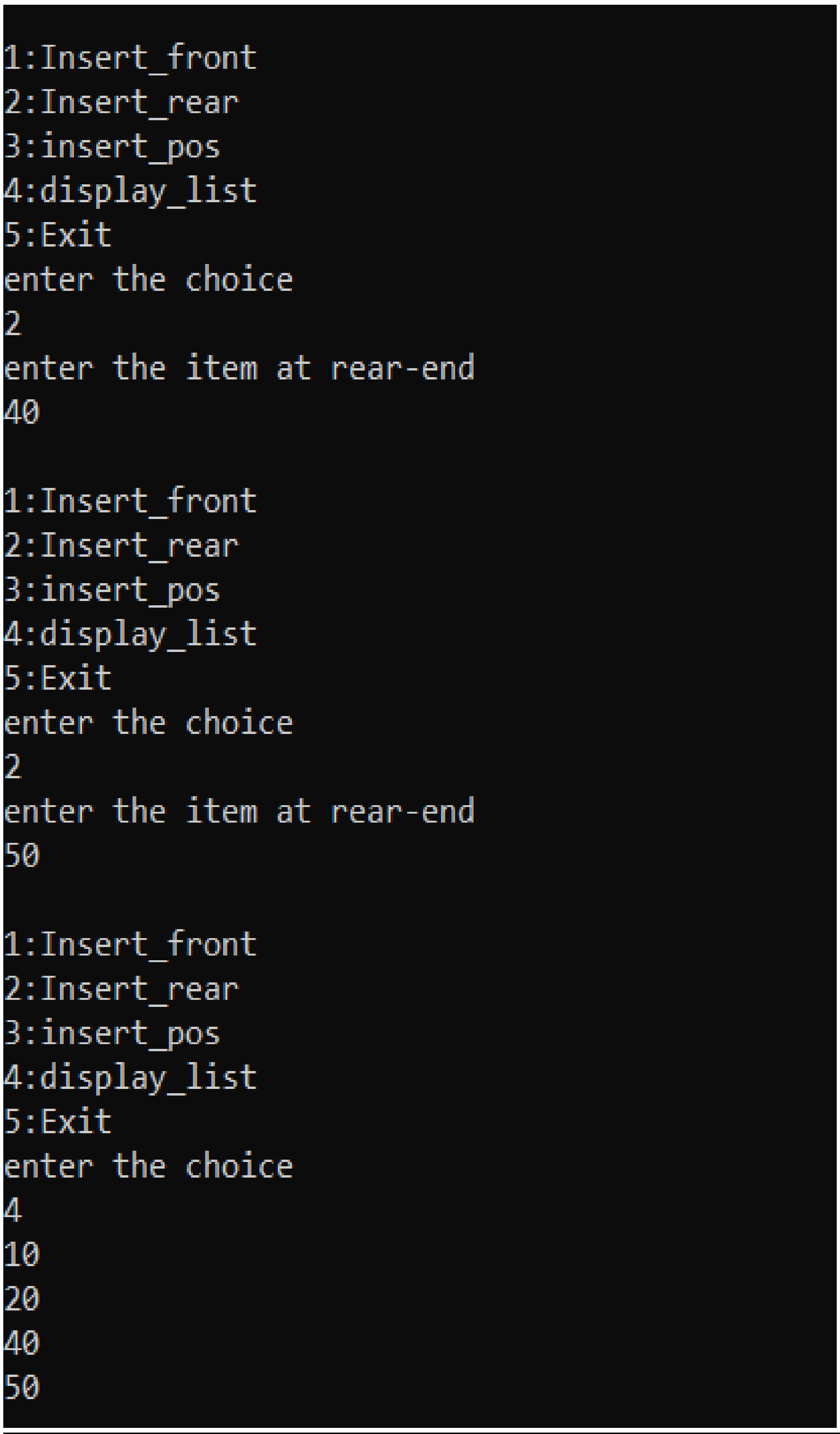
```
temp->link = NULL;
if (first == NULL && pos == 1)
return temp;
if (first == NULL)
{
printf("invalid pos\n");
return first;
}
if (pos == 1)
{
temp->link = first;
return temp;
}
count = 1;
prev = NULL;
cur = first;
while (cur != NULL && count != pos)
{
prev = cur;
cur = cur->link;
count++;
}
if (count == pos)
{
prev->link = temp;
temp->link = cur;
return first;
}
printf("IP\n");
return first;
}
void display(NODE first)
{
```

```
NODE temp;
if (first == NULL)
printf("list empty cannot display items\n");
for (temp = first; temp != NULL; temp = temp->link)
{
printf("%d\n", temp->info);
}
}
void main()
{
int item, choice, pos;
NODE first = NULL;
for (;;)
{
printf("\n1:Insert_front\n2:Insert_rear\n3:insert_pos\n4:display_li
st\n5:Exit\n");
printf("enter the choice\n");
scanf("%d", &choice);
switch (choice)
{
case 1:
printf("enter the item at front-end\n");
scanf("%d", &item);
first = insert_front(first, item);
break;
case 2:
printf("enter the item at rear-end\n");
scanf("%d", &item);
first = insert_rear(first, item);
break;
case 3:
printf("enter the position and item:\n");
scanf("%d", &pos);
scanf("%d",&item);
first = insert_pos(item, pos, first);
break;
case 4:
display(first);
break;
default:
exit(0);
}
}
}
```

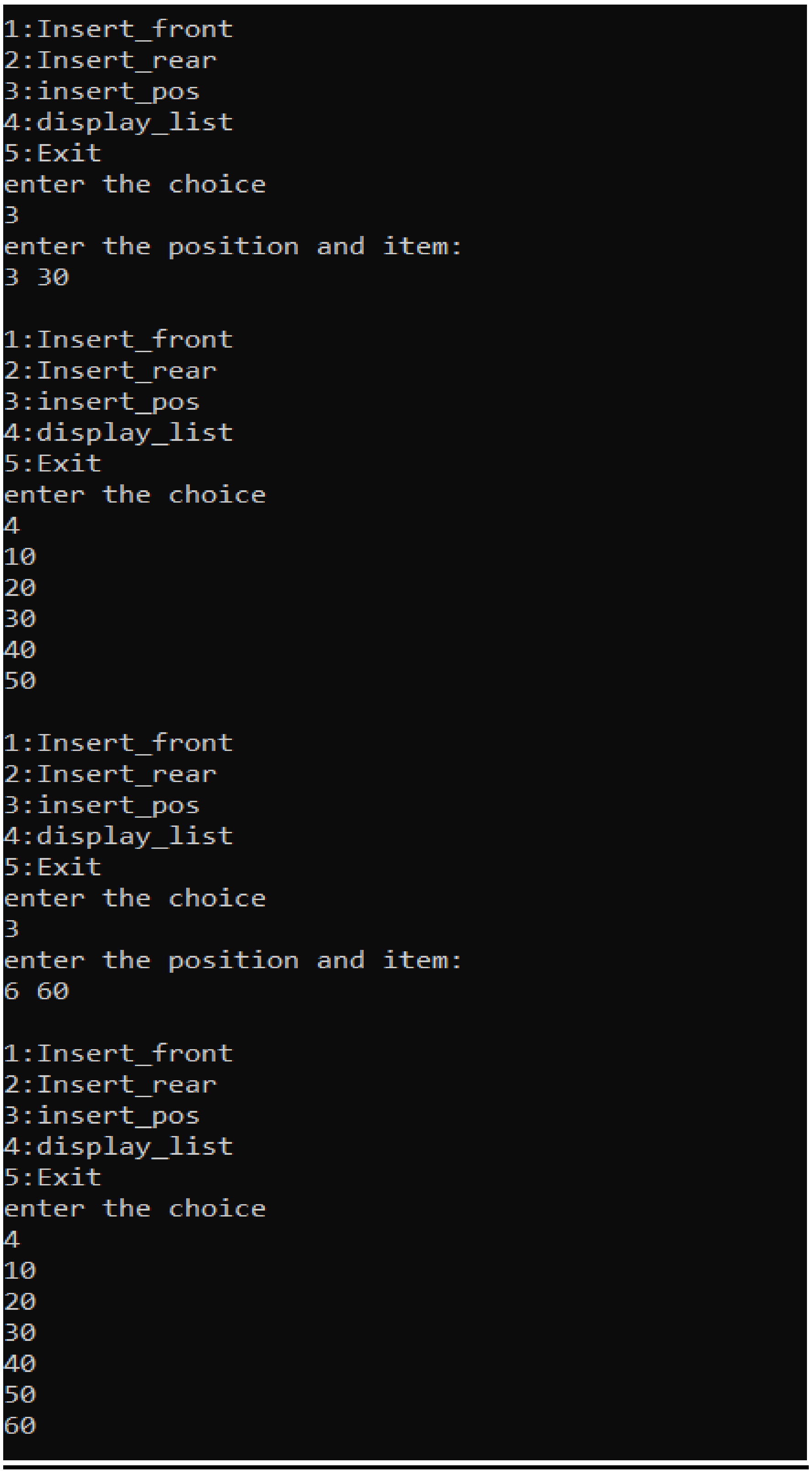
<u>OUTPUT :</u>



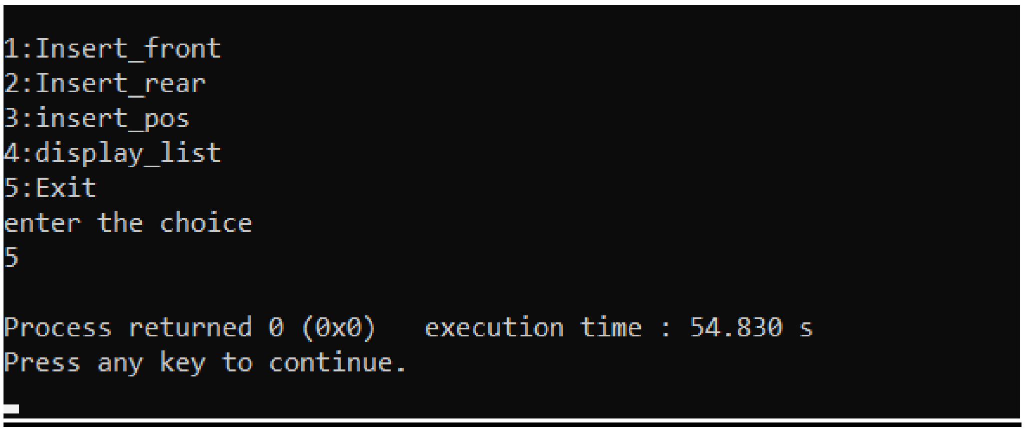
```
1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
1
enter the item at front-end
20

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
1
enter the item at front-end
10

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
4
10
20
```



```
1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
2
enter the item at rear-end
40

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
2
enter the item at rear-end
50

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
4
10
20
40
50
```



```
1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
3
enter the position and item:
3 30

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
4
10
20
30
40
50

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
3
enter the position and item:
6 60

1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
4
10
20
30
40
50
60
```



```
1:Insert_front
2:Insert_rear
3:insert_pos
4:display_list
5:Exit
enter the choice
5

Process returned 0 (0x0)   execution time : 54.830 s
Press any key to continue.
```