

Deletion

classmate

Date _____

Page _____

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

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

```
void create 0;
```

```
void display 0;
```

```
void delete - begin 0;
```

```
void delete - end 0;
```

```
void delete - pos 0;
```

```
struct node
```

```
{
```

```
int info;
```

```
struct node * next;
```

```
};
```

```
struct node * start = NULL
```

```
int main 0
```

```
{ int choice;
```

```
while(1) {
```

```
printf("\n\n 1. Create\n");
```

```
printf("\n 2. Display\n");
```

```
printf("\n 3. Delete from beginning\n");
```

```
printf("\n 4. Delete from the end\n");
```

```
printf("\n 5. Delete from specified position\n");
```

```
printf("\n 6. Exit\n");
```

```
printf("Enter your choice: ");
```

```
scanf("%d", &choice);
```

```
switch (choice)
```

```
{
```

case 1:

create 0;
break;

case 2: display 0;
break;

case 3: delete - begin 0;
break;

case 4:

delete - end 0;
break;

case 5: delete - pos 0;
break;

case 6:

exit(0);
break;

default

printf("n wrong choice: n");
break;

}

}

return 0;
} void create 0
{

```
struct node * temp, * ptr;  
temp = (struct node *) malloc (size of (struct node));  
if (temp == NULL)
```

```
{  
    printf ("\n Out of memory space: \n");  
    exit (0);  
}
```

```
{  
    printf ("\n Enter the data value for the node: \n");  
    scanf ("%d", &temp->info);  
    temp->next = NULL;  
    if (start == NULL)
```

```
{  
    start = temp;
```

```
}  
else
```

```
{  
    temp->next = start;  
    start = temp;
```

```
}
```

```
}
```

```
void display ()
```

```
{
```

```
    struct node * ptr;  
    if (start == NULL)
```

```
{
```

```
    printf ("\n list is empty: \n");  
    return;
```

}

else

{

ptr = start;

printf("\n The list elements are : \n");

while (ptr != NULL)

{

printf ("%d\t" ptr->info);

ptr = ptr->next

}

}

}

void delete - begin ()

{

struct node * ptr

if (ptr == NULL)

{

printf ("\n List is Empty: \n");

return;

}

else

{

ptr = start

while (ptr->next != NULL)

{

```
temp = ptr  
ptr = ptr -> next
```

```
}
```

```
temp -> next = NULL
```

```
printf("\n The deleted element is: %d\t", ptr->info);  
free(ptr);
```

```
}
```

```
}
```

```
void delete - pos 0
```

```
{
```

```
int i, pos;
```

```
struct node *temp, *ptr;
```

```
if (start == NULL)
```

```
{
```

```
printf("\n The List is Empty: \n");
```

```
return;
```

```
}
```

```
else
```

```
{
```

```
printf("\n Enter the position of the node to be  
deleted: \t");
```

```
scanf("%d", &pos);
```

```
if (pos == 0)
```

```
{
```

```
ptr = start
```

```
start = start -> next;
```

```
printf("\n The deleted element is: %d\t", ptr->info);
```

```
free(ptr);
```

```
}  
else  
{  
    ptr = start  
    for (i = 0; i pos; i++) { temp = ptr; ptr = ptr->next;  
        if (ptr == NULL
```

```
    } printf ("\n Position not found: \n");  
    return;
```

```
}
```

```
}
```

```
temp->next = ptr->next;  
printf ("\n The deleted element is: %d", d);  
ptr = ptr->next;  
if (ptr == NULL)
```

```
{
```

```
    printf ("\n Position not found: \n");  
    return;
```

```
}
```

```
}
```

```
temp->next = ptr->next;  
printf ("\n The deleted element is: %d", d);  
free(ptr);
```

```
}
```

```
}
```

```
}
```