

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

```
void create ();
void display ();
void insert - begin ();
void insert - end ();
void insert - pos ();
void delete - begin ();
void delete - end ();
void delete - pos ();
```

```
struct node
{
    int info;
    struct node *next;
};
struct node *start = NULL;
```

```
int main ()
```

```
{
```

```
    int choice;
```

```
    while (1) {
```

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

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

```
        printf ("3. Insert at the beginning\n");
```

```
        printf ("4. Insert at the end\n");
```

```
        printf ("5. Insert at specified position\n");
```

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

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

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

```
        switch (choice)
```

```
{
```


case 1:

create();

break;

case 2:

display();

break;

case 3:

insert_begin();

break;

case 4:

insert_end();

break;

case 5:

insert_pos();

break;

case 6:

exit(0);

~~break~~ break;

default

printf("\n Wrong choice: \n");

}

}

return 0;

}

void create()

{

struct node *temp, *ptr;

temp = (struct node *) malloc(sizeof(struct node));

if(temp == NULL)

{

printf("\n Out of memory space: \n");

exit(0)


```
}  
printf ("\n Enter the data value for the node: \t");  
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 The list elements are: \n");  
    while (ptr != NULL) return
```

```
{
```

```
    print else
```

```
{
```

```
    ptr = start  
    printf ("\n The list elements are: \n");  
    while (ptr != NULL)
```


classmate
Date _____
Page _____

```
{  
    printf("d\t", ptr->info);  
    ptr = ptr->next;  
}
```

```
}
```

```
}
```

```
}
```

```
void insert_begin()
```

```
{
```

```
    struct node * temp;
```

```
    temp = (struct node *) malloc (size of struct node)
```

```
    if (temp == NULL)
```

```
{
```

```
    printf("\n Out of memory space: \n");  
    return;
```

```
}
```

```
    printf("\n Enter the data value for the node: \t");
```

```
    scanf("%d", &temp->info);
```

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

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

```
{
```

```
        start = temp
```

```
}
```

```
    else
```

```
{
```

```
        temp->next = start
```

```
        start = temp
```

```
}
```


}

void insert - end 0

{

struct node *temp, *ptr;

temp = (struct node *) malloc (size of (struct node));

if (temp == NULL)

{

start = temp;

}

else

{

ptr = start;

while (ptr->next != NULL)

{

ptr = ptr->next;

}

ptr->next = temp

}

}

void insert - pos 0

{

struct node *ptr, *temp;

int i, pos;

temp = (struct node *) malloc (size of (struct node));

if (temp == NULL)

{

printf ("\n Enter the position for the new node to be inserted: ");

scanf


```
scanf ("%d", &pos);  
printf ("\n Enter the data value of the node:");  
scanf ("%d", &temp->info);
```

```
temp->next = NULL  
if (pos == 0)
```

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

```
}  
else
```

```
{  
    for (i=0, ptr = start; i < pos-1; i++) { ptr = ptr->next;  
        if (ptr == NULL
```

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

```
}
```

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

```
}
```

```
}
```