ANUSREE. P. R

AM. SC. P2CSC19011

**SINGLY LINKED LIST**

#include<iostream>

#include<cstdio>

#include<cstdlib>

using namespace std;

struct node

{

int info;

struct node \*next;

}\*start;

class single\_llist

{

public:

node\* create\_node(int);

void insert\_begin();

void insert\_pos();

void insert\_last();

void delete\_pos();

void display();

single\_llist()

{

start = NULL;

}

};

node \*single\_llist::create\_node(int value)

{

struct node \*temp, \*s;

temp = new(struct node);

if (temp == NULL)

{

cout<<"Memory not allocated "<<endl;

return 0;

}

else

{

temp->info = value;

temp->next = NULL;

return temp;

}

}

void single\_llist::insert\_begin()

{

int value;

cout<<"Enter the value to be inserted: ";

cin>>value;

struct node \*temp, \*p;

temp = create\_node(value);

if (start == NULL)

{

start = temp;

start->next = NULL;

}

else

{

p = start;

start = temp;

start->next = p;

}

cout<<"Element Inserted at beginning"<<endl;

}

void single\_llist::insert\_last()

{

int value;

cout<<"Enter the value to be inserted: ";

cin>>value;

struct node \*temp, \*s;

temp = create\_node(value);

s = start;

while (s->next != NULL)

{

s = s->next;

}

temp->next = NULL;

s->next = temp;

cout<<"Element Inserted at last"<<endl;

}

void single\_llist::insert\_pos()

{

int value, pos, counter = 0;

cout<<"Enter the value to be inserted: ";

cin>>value;

struct node \*temp, \*s, \*ptr;

temp = create\_node(value);

cout<<"Enter the postion at which node to be inserted: ";

cin>>pos;

int i;

s = start;

while (s != NULL)

{

s = s->next;

counter++;

}

if (pos == 1)

{

if (start == NULL)

{

start = temp;

start->next = NULL;

}

else

{

ptr = start;

start = temp;

start->next = ptr;

}

}

else if (pos > 1 && pos <= counter)

{

s = start;

for (i = 1; i < pos; i++)

{

ptr = s;

s = s->next;

}

ptr->next = temp;

temp->next = s;

}

else

{

cout<<"Positon out of range"<<endl;

}

}

void single\_llist::delete\_pos()

{

int pos, i, counter = 0;

if (start == NULL)

{

cout<<"List is empty"<<endl;

return;

}

cout<<"Enter the position of value to be deleted: ";

cin>>pos;

struct node \*s, \*ptr;

s = start;

if (pos == 1)

{

start = s->next;

}

else

{

while (s != NULL)

{

s = s->next;

counter++;

}

if (pos > 0 && pos <= counter)

{

s = start;

for (i = 1;i < pos;i++)

{

ptr = s;

s = s->next;

}

ptr->next = s->next;

}

else

{

cout<<"out of range"<<endl;

}

free(s);

cout<<"Element Deleted"<<endl;

}

}

void single\_llist::display()

{

struct node \*temp;

if (start == NULL)

{

cout<<"List is Empty"<<endl;

return;

}

temp = start;

cout<<"Elements of list are: "<<endl;

while (temp != NULL)

{

cout<<temp->info<<"--";

temp = temp->next;

}

}

main()

{

int choice, nodes, element, position, i;

single\_llist sl;

start = NULL;

while (1)

{

cout<<endl;

cout<<"1.Insert Node at beginning\t2.Insert node at last\t3.Insert node at position\n";

cout<<"4.Delete a Particular Node\t5.Display Linked List\n";

cout<<"Enter your choice : ";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Inserting Node at Beginning: "<<endl;

sl.insert\_begin();

cout<<endl;

break;

case 2:

cout<<"Inserting Node at Last: "<<endl;

sl.insert\_last();

cout<<endl;

break;

case 3:

cout<<"Inserting Node in the Middle:"<<endl;

sl.insert\_pos();

cout<<endl;

break;

case 4:

cout<<"Delete a particular node: "<<endl;

sl.delete\_pos();

break;

case 5:

cout<<"Display elements of link list"<<endl;

sl.display();

cout<<endl;

break;

default:

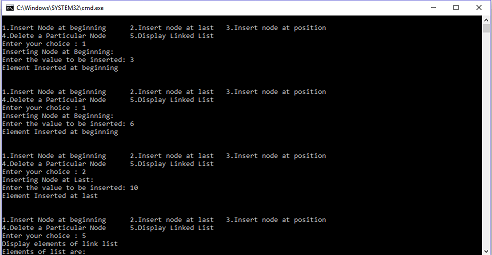
cout<<"Enter the correct input"<<endl;

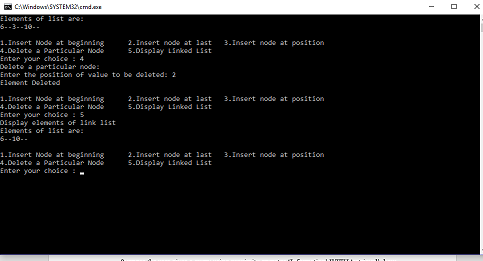
}

}

}

**OUTPUT**

****



**Github :-**