PROGRAM-NUMBER-1

*OBJECTIVE:-*To implement array and perform all the operation on Array.

*PROGRAM:-*

// OPERATION ON ARRAY

#include <stdio.h>

#define N 25

int a[N];

void traverse(int n);

void insert\_beg(int data, int n);

void insert\_end(int data, int n);

void insert\_pos(int data, int n, int pos);

void del\_beg(int n);

void del\_end(int n);

void del\_pos(int n, int pos);

void sort(int n);

int search(int n, int find);

void reverse(int n);

int main()

{

int n, i, choice, option, data, pos, find;

printf("Enter the element you want to enter in an array");

scanf("%d", &n);

for (i = 0; i < n; i++)

{

scanf("%d", &a[i]);

}

printf("Enter your choice\n");

printf("CHOICE 1:- TRAVERSING OF ARRAY\n");

printf("CHOICE 2:- INSERTION OF ARRAY\n");

printf("CHOICE 3:- DELETION OF ARRAY\n");

printf("CHOICE 4:- SORTING OF ARRAY\n");

printf("CHOICE 5:- SEARCHING OF ARRAY\n");

printf("CHOICE 6:- REVERSE OF ARRAY\n");

printf("CHOICE 7:- EXIT OF ARRAY\n");

do

{

scanf("%d", &choice);

switch (choice)

{

case 1:

traverse(n);

break;

case 2:

{

printf("Enter option where you want insertion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-POSITION\n");

printf("OPTION 4:-EXIT\n");

do

{

scanf("%d", &option);

printf("Enter the data you want to insert:\n");

scanf("%d", &data);

switch (option)

{

case 1:

insert\_beg(data, n);

break;

case 2:

insert\_end(data, n);

break;

case 3:

{

printf("Enter the position where you want insertion:\n");

scanf("%d", &pos);

insert\_pos(data, n, pos);

}

break;

case 4:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 4);

}

break;

case 3:

{

printf("Enter option where you want deletion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-POSITION\n");

printf("OPTION 4:-EXIT\n");

do

{

scanf("%d", &option);

switch (option)

{

case 1:

del\_beg(n);

break;

case 2:

del\_end(n);

break;

case 3:

{

printf("Enter the position where you want deletion:\n");

scanf("%d", &pos);

del\_pos(n, pos);

}

break;

case 4:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 4);

}

break;

case 4:

sort(n);

break;

case 5:

{

printf("Enter the element you want to search:\n");

scanf("%d", &find);

search(n, find);

}

break;

case 6:

reverse(n);

break;

case 7:

break;

default:

printf("Please, enter the correct option\n");

break;

}

} while (choice != 7);

}

void traverse(int n)

{

int i;

for (i = 0; i < n; i++)

{

printf("%d\n", a[i]);

}

}

void insert\_beg(int data, int n)

{

int i;

printf("insertion at Beginning\n");

if (n == N)

{

printf("Array is overflow\n");

}

else

{

for (i = n; i >= 0; i--)

{

a[i + 1] = a[i];

}

n = n + 1;

a[0] = data;

traverse(n);

}

}

void insert\_end(int data, int n)

{

printf("insertion at end\n");

if (n == N)

{

printf("Array is overflow\n");

}

else

{

a[n] = data;

n = n + 1;

traverse(n);

}

}

void insert\_pos(int data, int n, int pos)

{

int i;

printf("insertion at end");

if (n == N)

{

printf("Array is overflow");

}

else

{

for (i = n; i >= pos - 1; i--)

{

a[i + 1] = a[i];

}

n = n + 1;

a[pos - 1] = data;

traverse(n);

}

}

void del\_beg(int n)

{

int i;

printf("Deletion at beginning\n");

if (n == 0)

{

printf("Array is underflow\n");

}

else

{

for (i = 0; i < n; i++)

{

a[i] = a[i + 1];

}

n--;

traverse(n);

}

}

void del\_end(int n)

{

printf("Deletion at end\n");

if (n == 0)

{

printf("Array is underflow\n");

}

else

{

a[n] = '\0';

n = n - 1;

traverse(n);

}

}

void del\_pos(int n, int pos)

{

int i;

printf("Deletion at position\n");

if (n == 0)

{

printf("Array is underflow\n");

}

else

{

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

{

a[i] = a[i + 1];

}

n = n - 1;

traverse(n);

}

}

void sort(int n)

{

printf("Sorted array is:\n");

int i = 0, t;

while (i < n)

{

for (int j = i + 1; j < n; j++)

{

if (a[i] > a[j])

{

t = a[i];

a[i] = a[j];

a[j] = t;

}

}

i++;

}

traverse(n);

}

int search(int n, int find)

{

int i = 0;

while (i < n)

{

if (find == a[i])

{

printf("Yehh!!, Element found\n");

return 1;

}

i++;

}

printf("Oops, Element not found!!\n");

return 0;

}

void reverse(int n)

{

printf("Reverse of an array\n");

int i;

for (i = n - 1; i >= 0; i--)

{

printf("%d\t", a[i]);

}

}

*OUTPUT:-*

Enter the element you want to enter in an array5

1 2 3 4 5

Enter your choice

CHOICE 1:- TRAVERSING OF ARRAY

CHOICE 2:- INSERTION OF ARRAY

CHOICE 3:- DELETION OF ARRAY

CHOICE 4:- SORTING OF ARRAY

CHOICE 5:- SEARCHING OF ARRAY

CHOICE 6:- REVERSE OF ARRAY

CHOICE 7:- EXIT OF ARRAY

1

1

2

3

4

5

4

Sorted array is:

1

2

3

4

5

3

Enter option where you want deletion

OPTION 1:-BEGINING

OPTION 2:-END

OPTION 3:-POSITION

OPTION 4:-EXIT

3

Enter the position where you want deletion:

2

Deletion at position

1

2

4

5

4

6

Reverse of an array

0 5 4 2 1 5

Enter the element you want to search:

7

Oops, Element not found!!

PROGRAM-NUMBER-2

*OBJECTIVE:-* To implement stack using array.

*PROGRAM:-*

#include<stdio.h>

#define MAX 10

void push();

void pop();

void peep();

void travese();

int stack[MAX], top=-1;

int main()

{

int choice;

do

{

printf("\nEnter your choice:\n");

printf("1.PUSH\n");

printf("2.POP\n");

printf("3.TRAVERSE\n");

printf("4.PEEP\n");

printf("5.EXIT\n");

scanf("%d",&choice);

switch (choice)

{

case 1:

push();

break;

case 2:

pop();

break;

case 3:

travese();

break;

case 4:

peep();

break;

case 5:

printf("End of program\n");

break;

default:

printf("Enter the coorect choice");

break;

}

} while (choice!=5);

}

void push()

{

int data;

if (top==(MAX-1))

{

printf("overflow");

}

else

{

printf("enter the data\n");

scanf("%d",&data);

top++;

stack[top]=data;

}

}

void travese()

{

for (int i = top; i > -1; i--)

{

printf("%d\t", stack[i]);

}

}

void pop()

{

if (top == -1)

{

printf("STACK UNDERFLOW\n");

}

else

{

printf("Deleted element is : %d", stack[top]);

top--;

}

}

void peep()

{

printf("%d", stack[top]);

}

*OUTPUT:-*

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

1

enter the data

25

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

1

enter the data

26

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

1

enter the data

27

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

3

27 26 25

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

4

27

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

2

Deleted element is : 27

Enter your choice:

1.PUSH

2.POP

3.TRAVERSE

4.PEEP

5.EXIT

5

End of program

PROGRAM-NUMBER-3

*OBJECTIVE:-* To implement stack using Linked list.

*PROGRAM:-*

// STACK USING LINKED LIST

#include <stdio.h>

#include <stdlib.h>

void push();

void display();

void pop();

void peep();

struct stack

{

int data;

struct stack \*link;

} \*top = '\0';

typedef struct stack stack;

int main()

{

int choice;

do

{

printf("Enter your choice\n");

printf("1.PUSH\n");

printf("2.POP\n");

printf("3.DISPLAY\n");

printf("4.PEEP\n");

printf("5.EXIT\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

push();

break;

case 2:

// pop();

break;

case 3:

display();

break;

case 4:

peep();

break;

case 5:

printf("End of the program");

break;

default:

printf("Please Enter the correct opton:\n");

break;

}

} while (choice != 5);

}

void push()

{

stack \*t;

int data;

t = (stack \*)malloc(sizeof(stack));

printf("Enter the data:\n");

scanf("%d", &t->data);

t->link = top;

top = t;

}

void display()

{

stack \*t;

t = top;

printf("Display of a Stack\n");

while (t!='\0')

{

printf("%d\t", t->data);

t = t->link;

}

}

void pop()

{

stack \*t;

t=top;

printf("Deleted element is %d",t->data);

free(t);

}

void peep()

{

stack \*t;

t=top;

printf("%d is at top of the stack",t->data);

}

*OUTPUT:-*

Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

1

Enter the data:

10

Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

1

Enter the data:

11

Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

1

Enter the data:

12

Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

3

Display of a Stack

12 11 10 Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

2

Enter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

4

12 is at top of the stackEnter your choice

1.PUSH

2.POP

3.DISPLAY

4.PEEP

5.EXIT

5

End of the program

PROGRAM-NUMBER-4

*OBJECTIVE:-* To implement singly linked list.

*PROGRAM:-*

// SINGLY LINKED LIST

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

} \*head = '\0';

typedef struct node node;

void create();

void traverse();

void insert\_beg(int info);

void insert\_end(int info);

void insert\_sort(int info);

void insert\_pos(int info, int pos);

void del\_beg();

void del\_end();

void del\_pos(int pos);

int search(int find);

void reverse();

int main()

{

int choice, option, info, pos, find;

create();

printf("\nEnter your choice\n");

printf("\nCHOICE 1:- TRAVERSING OFLINKED LIST\n");

printf("\nCHOICE 2:- INSERTION OFLINKED LIST\n");

printf("\nCHOICE 3:- DELETION OFLINKED LIST\n");

printf("\nCHOICE 4:- SEARCHING OFLINKED LIST\n");

printf("\nCHOICE 5:- REVERSE OF LINKED LIST\n");

printf("\nCHOICE 6:- EXIT OF LINKED LIST\n");

do

{

scanf("%d", &choice);

switch (choice)

{

case 1:

traverse();

break;

case 2:

{

printf("\nEnter option where you want insertion\n");

printf("\nOPTION 1:-BEGINING\n");

printf("\nOPTION 3:-POSITION\n");

printf("\nOPTION 4:-SORTED POSITION\n");

printf("\nOPTION 5:-EXIT\n");

do

{

scanf("%d", &option);

printf("Enter the data you want to insert:\n");

scanf("%d", &info);

switch (option)

{

case 1:

insert\_beg(info);

break;

case 2:

insert\_end(info);

break;

case 3:

{

printf("Enter the data of node after which you want insertion:\n");

scanf("%d", &pos);

insert\_pos(info, pos);

}

break;

case 4:

insert\_sort(info);

break;

case 5:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 5);

}

break;

case 3:

{

printf("\nEnter option where you want deletion\n");

printf("\nOPTION 1:-BEGINING\n");

printf("\nOPTION 2:-END\n");

printf("\nOPTION 3:-POSITION\n");

printf("\nOPTION 4:-EXIT\n");

do

{

scanf("%d", &option);

switch (option)

{

case 1:

del\_beg();

break;

case 2:

del\_end();

break;

case 3:

{

printf("Enter the data of node which you want delete:\n");

scanf("%d", &pos);

del\_pos(pos);

}

break;

case 4:

break;

default:

printf("Please,Enter the correct option!!\n");

break;

}

} while (option != 4);

}

break;

case 4:

{

printf("Enter the element you want to search:\n");

scanf("%d", &find);

search(find);

}

break;

case 5:

reverse();

break;

case 6:

break;

default:

printf("Please, enter the correct option\n");

break;

}

} while (choice != 6);

}

void create()

{

node \*p, \*c;

char ch;

p = (node \*)malloc(sizeof(node));

head = p;

printf("Enter the first element of list:-\n");

scanf("%d", &p->data);

p->next = '\0';

do

{

c = (struct node \*)malloc(sizeof(struct node));

printf("Enter the next element of the list:-\n");

scanf("%d", &c->data);

p->next = c;

p = c;

printf("Do you want next node!!\n Choose:-Y/N\n");

scanf(" %c", &ch);

} while (ch == 'y' || ch == 'Y');

p->next = '\0';

}

void traverse()

{

node \*p;

p = head;

while (p != '\0')

{

printf("|%d|%u|->", p->data, p->next);

p = p->next;

}

}

void insert\_beg(int info)

{

node \*p;

p = (node \*)malloc(sizeof(node));

p->data = info;

p->next = head;

head = p;

traverse();

}

void insert\_end(int info)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->next != '\0')

{

c = c->next;

}

c->next = p;

p->next = '\0';

traverse();

}

void insert\_pos(int info, int pos)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->data != pos)

{

c = c->next;

}

p->next = c->next;

c->next = p;

traverse();

}

void insert\_sort(int info)

{

node \*p, \*c, \*temp;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->data <= p->data)

{

c = c->next;

}

p->next = c->next;

c->next = p;

traverse();

}

void del\_beg()

{

node \*p;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

head = p->next;

free(p);

traverse();

}

}

void del\_end()

{

node \*p, \*c;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

while (p->next != '\0')

{

c = p;

p = p->next;

}

c->next = '\0';

free(p);

traverse();

}

}

void del\_pos(int pos)

{

node \*p, \*c;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

while (p->data != pos)

{

c = p;

p = p->next;

}

c->next = p->next;

free(p);

traverse();

}

}

int search(int find)

{

node \*p;

p = head;

while (p != '\0')

{

if (p->data == find)

{

printf("Element found!!!\n");

return 1;

}

p = p->next;

}

printf("Oops, Element does not exist!!\n");

return 0;

}

void reverse()

{

node \*p, \*c, \*t;

p = head;

t = '\0';

while (p != '\0')

{

c = p->next;

p->next = t;

t = p;

p = c;

}

traverse();

}

*OUTPUT:-*

Enter the first element of list:-

1

Enter the next element of the list:-

2

Do you want next node!!

Choose:-Y/N

y

Enter the next element of the list:-

3

Do you want next node!!

Choose:-Y/N

N

Enter your choice

CHOICE 1:- TRAVERSING OFLINKED LIST

CHOICE 2:- INSERTION OFLINKED LIST

CHOICE 3:- DELETION OFLINKED LIST

CHOICE 4:- SEARCHING OFLINKED LIST

CHOICE 5:- REVERSE OF LINKED LIST

CHOICE 6:- EXIT OF LINKED LIST

1

|1|11146168|->|2|11146184|->|3|0|->2

Enter option where you want insertion

OPTION 1:-BEGINING

OPTION 2:-END

OPTION 3:-POSITION

OPTION 4:-SORTED POSITION

OPTION 5:-EXIT

1

Enter the data you want to insert:

0

|0|11146152|->|1|11146168|->|2|11146184|->|3|0|->2

Enter the data you want to insert:

4

|0|11146152|->|1|11146168|->|2|11146184|->|3|11146216|->|4|0|->3

Enter the data you want to insert:

5

Enter the data of node after which you want insertion:

2

|0|11146152|->|1|11146168|->|2|11144680|->|5|11146184|->|3|11146216|->|4|0|->4

PROGRAM-NUMBER-5

*OBJECTIVE:-* To implement doubly linked list.

*PROGRAM:-*

// DOUBLY LINKED LIST

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

struct node \*pre;

} \*head = '\0';

typedef struct node node;

void create();

void traverse();

void reverse();

void insert\_beg(int info);

void insert\_end(int info);

void del\_beg();

void del\_end();

int main()

{

int choice, option, info;

create();

printf("Enter your choice\n");

printf("CHOICE 1:- TRAVERSING OF LINKED LIST\n");

printf("CHOICE 2:- REVERSE OF LINKED LIST\n");

printf("CHOICE 3:- INSERTION OF LINKED LIST\n");

printf("CHOICE 4:- DELETION OF LINKED LIST\n");

printf("CHOICE 5:- EXIT OF LINKED LIST\n");

do

{

scanf("%d", &choice);

switch (choice)

{

case 1:

traverse();

break;

case 2:

reverse();

break;

case 3:

{

printf("Enter option where you want insertion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-EXIT\n");

do

{

scanf("%d", &option);

printf("Enter the data you want to insert:\n");

scanf("%d", &info);

switch (option)

{

case 1:

insert\_beg(info);

break;

case 2:

insert\_end(info);

break;

case 3:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 3);

}

break;

case 4:

{

printf("Enter option where you want deletion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-EXIT\n");

do

{

scanf("%d", &option);

switch (option)

{

case 1:

del\_beg();

break;

case 2:

del\_end();

break;

case 3:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 3);

}

break;

case 5:

break;

default:

printf("Please, enter the correct option\n");

break;

}

} while (choice != 5);

}

void create()

{

node \*p, \*c;

char ch;

p = (node \*)malloc(sizeof(node));

head = p;

printf("Enter the first element of list:-\n");

scanf("%d", &p->data);

p->pre='\0';

head=p;

do

{

c = (struct node \*)malloc(sizeof(struct node));

printf("Enter the next element of the list:-\n");

scanf("%d", &c->data);

p->next = c;

c->pre = p;

p = c;

printf("Do you want next node!!\n Choose:-Y/N\n");

scanf(" %c", &ch);

} while (ch == 'y' || ch == 'Y');

p->next = '\0';

}

void traverse()

{

node \*p;

p = head;

while (p != '\0')

{

printf("%d\t", p->data);

p = p->next;

}

}

void reverse()

{

node \*p;

p = head;

while (p->next != '\0')

{

p = p->next;

}

while (p != '\0')

{

printf("%d\t",p->data);

p = p->pre;

}

}

void insert\_beg(int info)

{

node \*p;

p = (node \*)malloc(sizeof(node));

p->data = info;

p->next = head;

p->pre = '\0';

head= p;

traverse();

}

void insert\_end(int info)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->next != head)

{

c = c->next;

}

c->next = p;

p->pre= c;

p->next = '\0';

traverse();

}

void del\_beg()

{

node \*p, \*c = head;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

head = p->next;

head->pre='\0';

free(p);

traverse();

}

}

void del\_end()

{

node \*p, \*c;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

while (p->next != head)

{

c = p;

p = p->next;

}

c->next = '\0';

free(p);

traverse();

}

}

*OUTPUT:-*

Enter the first element of list:-

10

Enter the next element of the list:-

11

Do you want next node!!

Choose:-Y/N

y

Enter the next element of the list:-

12

Do you want next node!!

Choose:-Y/N

n

Enter your choice

CHOICE 1:- TRAVERSING OF LINKED LIST

CHOICE 2:- REVERSE OF LINKED LIST

CHOICE 3:- INSERTION OF LINKED LIST

CHOICE 4:- DELETION OF LINKED LIST

CHOICE 5:- EXIT OF LINKED LIST

2

12 11 10

PROGRAM-NUMBER-6

*OBJECTIVE:-* To implement circular linked list.

*PROGRAM:-*

// CIRCULAR LINKED LIST

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

} \*head = '\0';

typedef struct node node;

void create();

void traverse();

void insert\_beg(int info);

void insert\_end(int info);

void insert\_sort(int info);

void insert\_pos(int info, int pos);

void del\_beg();

void del\_end();

int main()

{

int choice, option, info, pos;

create();

printf("Enter your choice\n");

printf("CHOICE 1:- TRAVERSING OF LINKED LIST\n");

printf("CHOICE 2:- INSERTION OF LINKED LIST\n");

printf("CHOICE 3:- DELETION OF LINKED LIST\n");

printf("CHOICE 4:- EXIT OF LINKED LIST\n");

do

{

scanf("%d", &choice);

switch (choice)

{

case 1:

traverse();

break;

case 2:

{

printf("Enter option where you want insertion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-POSITION\n");

printf("OPTION 4:-EXIT\n");

do

{

scanf("%d", &option);

printf("Enter the data you want to insert:\n");

scanf("%d", &info);

switch (option)

{

case 1:

insert\_beg(info);

break;

case 2:

insert\_end(info);

break;

case 3:

{

printf("Enter the data of node after which you want insertion:\n");

scanf("%d", &pos);

insert\_pos(info, pos);

}

break;

case 4:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 4);

}

break;

case 3:

{

printf("Enter option where you want deletion\n");

printf("OPTION 1:-BEGINING\n");

printf("OPTION 2:-END\n");

printf("OPTION 3:-EXIT\n");

do

{

scanf("%d", &option);

switch (option)

{

case 1:

del\_beg();

break;

case 2:

del\_end();

break;

case 3:

break;

default:

printf("Please,Enter the correct option!!");

break;

}

} while (option != 3);

}

break;

case 4:

break;

default:

printf("Please, enter the correct option\n");

break;

}

} while (choice != 4);

}

void create()

{

node \*p, \*c;

char ch;

p = (node \*)malloc(sizeof(node));

head = p;

printf("Enter the first element of list:-\n");

scanf("%d", &p->data);

do

{

c = (struct node \*)malloc(sizeof(struct node));

printf("Enter the next element of the list:-\n");

scanf("%d", &c->data);

p->next = c;

p = c;

printf("Do you want next node!!\n Choose:-Y/N\n");

scanf(" %c", &ch);

} while (ch == 'y' || ch == 'Y');

p->next = head;

}

void traverse()

{

node \*p;

p = head;

do

{

printf("|%d|%u|->", p->data, p->next);

p = p->next;

} while (p != head);

printf("\b\b");

}

void insert\_beg(int info)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

p->next = head;

c = head;

while (c->next != head)

{

c = c->next;

}

head = p;

c->next = head;

traverse();

}

void insert\_end(int info)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->next != head)

{

c = c->next;

}

c->next = p;

p->next = head;

traverse();

}

void insert\_pos(int info, int pos)

{

node \*p, \*c;

p = (node \*)malloc(sizeof(node));

p->data = info;

c = head;

while (c->data != pos)

{

c = c->next;

}

p->next = c->next;

c->next = p;

traverse();

}

void del\_beg()

{

node \*p, \*c = head;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

while (c->next != head)

{

c = c->next;

}

head = p->next;

c->next = head;

free(p);

traverse();

}

}

void del\_end()

{

node \*p, \*c;

if (head = '\0')

{

printf("Underflow\n");

}

else

{

p = head;

while (p->next != head)

{

c = p;

p = p->next;

}

c->next = head;

free(p);

traverse();

}

}

*OUTPUT:-*

Enter the first element of list:-

21

Enter the next element of the list:-

22

Do you want next node!!

Choose:-Y/N

y

Enter the next element of the list:-

23

Do you want next node!!

Choose:-Y/N

n

Enter your choice

CHOICE 1:- TRAVERSING OF LINKED LIST

CHOICE 2:- INSERTION OF LINKED LIST

CHOICE 3:- DELETION OF LINKED LIST

CHOICE 4:- EXIT OF LINKED LIST

1

|21|6689720|->|22|6689736|->|23|6689704|2>

Enter option where you want insertion

OPTION 1:-BEGINING

OPTION 2:-END

OPTION 3:-POSITION

OPTION 4:-EXIT

1

Enter the data you want to insert:

20

|20|6689704|->|21|6689720|->|22|6689736|->|23|6689752|3>

Enter the data you want to insert:

24

Enter the data of node after which you want insertion:

22

|20|6689704|->|21|6689720|->|22|6689768|->|24|6689736|->|23|6689752|3>

Enter the data you want to insert:

4

Enter the data of node after which you want insertion:

23

|20|6689704|->|21|6689720|->|22|6689768|->|24|6689736|->|23|6688232|->|4|6689752|4>

Enter the data you want to insert:

4

3

Enter option where you want deletion

OPTION 1:-BEGINING

OPTION 2:-END

OPTION 3:-EXIT

2