29/01/2024

1. 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, Display contents of linked list.

c) Deletion of first element, specified element and last element in the list.

```
struct node
{
    int data;
    struct node *next;
};
Struct node *start = NULL;
Struct node *create_ll(struct node *start)
{
  struct node *new_node, *ptr;
  int num;
  printf("\n enter -1 to end");
  printf("\n enter the data:");
  scanf("%d", &num);
  while (num != -1)
  {
      new_node = (struct node*) malloc(sizeof(struct node));
      new_node -> data = num;
      if (start == NULL)
      {
          new_node -> next = NULL;
          start = new_node
      }
      else
      {
          ptr = start;
          while (ptr -> next != NULL)
              ptr = ptr -> next;
          ptr -> next = new_node
```

```
        printf("\n enter the data:");
        scanf("%d",&num);
    }
    return start;
};

struct node *display(struct node *start)
{
    struct node *ptr;
    ptr = start;
    while(ptr!=NULL)
    {
        printf("%d",ptr->data);
        ptr = ptr->next;
    }
    return start;
};

struct node *insert_beg(struct node *start)
{
    struct node *new_node;
    int num;
    printf("enter the data:");
    scanf("%d",&num);
    new_node = (struct node*) malloc(sizeof(struct node));
    new_node->data = num;
    new_node->next = start;
    start = new_node;
    return start;
};

struct node *insert_end(struct node *start)
{
    struct node *new_node, *ptr;
    int num;
    printf("enter the data:");
    scanf("%d",&num);
    new_node = (struct node*) malloc(sizeof(struct node));
    if(start==NULL
```

```
        new_node->next = NULL;
        start = new_node;
    }
    else
    {
        ptr = start;
        while (ptr->next != NULL)
        {
            ptr = ptr->next;
        }
        ptr->next = new_node;
    }
}

struct node *insert_before (struct node *start)
{
    struct node *new_node, *ptr, *preptr;
    int num, val;
    printf ("enter the data:");
    scanf ("%d", &num);
    printf ("enter the value before which data has to be inserted:");
    scanf ("%d", &val);
    new_node = (struct node*) malloc (sizeof (struct node));
    new_node -> data = num;
    ptr = start;
    while (ptr->data != val)
    {
        preptr = ptr;
        ptr = ptr->next;
    }
    preptr->next = new_node;
    new_node->next = ptr;
    return start;
};

struct node *insert_after (struct node *start)
{
    while (preptr->data != val)
    {
        preptr = ptr;
        ptr = ptr->next;
```

```
    preptr -> next = new_node;
    new_node -> next = ptr;
    return start;
  };

Struct node *delete_beg (struct node *start)
 {
    Struct node *ptr;
    ptr = start;
    start = start -> next;
    free (ptr);
    return start;
  };

Struct node *delete_end (struct node *start)
 {
    Struct node *ptr, *preptr;
    ptr = start;
    while (ptr -> next != NULL);
    {
        preptr = ptr;
        ptr = ptr -> next;
    }
    preptr -> next = NULL;
    free (ptr);
    return start;
  };

struct node *delete_node (struct node *start)
 {
   struct node *ptr, *preptr;
    int val;
    printf ("enter the value to be deleted:");
    scanf ("%d", &val);
    ptr = start;
    if (ptr -> data == val)
    {
        start = delete_beg (start);
        return start;
    }
```

```
else{
  while (ptr->data!=start);
  {
    preptr = ptr;
    ptr = ptr->next;
  }
  preptr->next = ptr->next;
  free(ptr);
  return start;
}
};
```

op:

---menu---

1. create linked list
2. display
3. insert-beg
4. insert-end
5. insert before
6. insert after
7. del-beg
8. del-end
9. del-node
10. exit

enter the choice

1

Enter -1 to exit

Enter the number: 40

Enter the number: 20

Enter the number: -1

linked list created -

enter the choice:

2

40 20

enter choice : 3

10

enter choice : 2

10 40 20

enter choice : 4

30

enter choice : 2

10 40 20 30

---

enter choice : 7

deleted element : 10

enter choice : 2

40 20 30

enter choice : 8

deleted element : 30

enter choice : 2

40 20

enter choice : 9

enter element : 40

enter choice : 2

20

write o/p completely

22/1/24

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

struct node
{
    int data;
    struct node *next;
};

struct node *head = NULL;

struct node *create_ll(struct node *head)
{
    struct node *new_node, *ptr;
    int num;
    printf("Enter -1 to exit.. \n");
    printf("Enter the num: ");
    scanf("%d", &num);

    while (num != -1)
    {
        new_node = (struct node *)malloc(sizeof(struct node));
        new_node->data = num;
        if (head == NULL)
        {
            head = new_node;
            new_node->next = NULL;
        }
        else
        {
            ptr = head;
            while (ptr->next != NULL)
            {
                ptr = ptr->next;
            }
            ptr->next = new_node;
            new_node->next = NULL;
```

```c
            new_node->next = NULL;
        }
        printf("Enter the num: ");
        scanf("%d", &num);
    }
    return head;
}

struct node *insert_beg(struct node *head)
{
    struct node *new_node;
    int num;
    printf("Enter the num: ");
    scanf("%d", &num);
    new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = num;
    if (head == NULL)
    {
        head = new_node;
        new_node->next = NULL;
    }
    else
    {
        new_node->next = head;
        head = new_node;
    }
    return head;
}

struct node *insert_end(struct node *head)
{
    struct node *ptr, *new_node;
    int num;
    printf("Enter the num: ");
    scanf("%d", &num);
    new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = num;
```

```
74      new_node->next = NULL;
75      if (head == NULL)
76      {
77          head = new_node;
78      }
79      else
80      {
81          ptr = head;
82          while (ptr->next != NULL)
83          {
84              ptr = ptr->next;
85          }
86          ptr->next = new_node;
87      }
88      return head;
89  }
90
91  struct node *insert_before(struct node *head)
92  {
93      struct node *new_node, *ptr, *prevptr;
94      int num, val;
95      printf("Enter the num: ");
96      scanf("%d", &num);
97      printf("Enter the value before which number has to be inserted: ");
98      scanf("%d", &val);
99      new_node = (struct node *)malloc(sizeof(struct node));
100     new_node->data = num;
101     ptr = head;
102     while (ptr->next != NULL)
103     {
104         prevptr = ptr;
105         ptr = ptr->next;
106     }
107     prevptr->next = new_node;
108     new_node->next = ptr;
109     return head;
110 }
```

```c
}
struct node *insert_after(struct node *head)
{
    struct node *new_node, *ptr, *prevptr;
    int num, val;
    printf("Enter the num: ");
    scanf("%d", &num);
    printf("Enter the value before which number has to be inserted: ");
    scanf("%d", &val);
    new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = num;
    ptr = head;
    while (ptr->next != NULL)
    {
        prevptr = ptr;
        ptr = ptr->next;
    }
    prevptr = ptr;
    ptr = ptr->next;
    prevptr->next = new_node;
    new_node->next = ptr;
    return head;
}
struct node *display(struct node *head)
{
    struct node *ptr;
    if (head == NULL)
    {
        printf("Linked List is empty...\n");
    }
    else
    {
        ptr = head;
        while (ptr != NULL)
        {
            printf("%d ", ptr->data);
            ptr = ptr->next;
```

```
147             }
148             printf("\n");
149         }
150         return head;
151     }
152
153     struct node *delete_beg(struct node *head)
154     {
155         struct node *ptr;
156         if (head == NULL)
157         {
158             printf("Nothing to delete.. \n");
159         }
160         else
161         {
162             ptr = head;
163             head = ptr->next;
164             free(ptr);
165         }
166         return head;
167     }
168
169     struct node *delete_end(struct node *head)
170     {
171         struct node *ptr, *prevptr;
172         ptr = head;
173         while (ptr->next != NULL)
174         {
175             prevptr = ptr;
176             ptr = ptr->next;
177         }
178         prevptr->next = NULL;
179         free(ptr);
180         return head;
181     }
182
```

```c
struct node *delete_node(struct node *head)
{
    struct node *ptr, *prevptr;
    int val;
    printf("Enter the value that has to be deleted: ");
    scanf("%d", &val);
    ptr = head;
    if (ptr->data == val)
    {
        head = delete_beg(head);
        return head;
    }
    else
    {
        while (ptr->data != val)
        {
            prevptr = ptr;
            ptr = ptr->next;
        }
        prevptr->next = ptr->next;
        free(ptr);
        return head;
    }
}

int main()
{
    int choice;
    printf("---------menu-------\n");
    printf("1.create lined list\n 2.display\n 3.insert_beg\n 4.insert_end\n 5.insert_before\n 6.insert_after\n 7.del_beg\n 8.del
    do
    {
        printf("enter the choice:\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
```

```c
    do
    {
        printf("enter the choice:\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            head = create_ll(head);
            printf("linked list created");
            break;
        case 2:
            head = display(head);
            break;
        case 3:
            head = insert_beg(head);
            break;
        case 4:
            head = insert_end(head);
            break;
        case 5:
            head = insert_before(head);
            break;
        case 6:
            head = insert_after(head);
            break;
        case 7:
            head = delete_beg(head);
            break;
        case 8:
            head = delete_end(head);
            break;
        case 9:
            head = delete_node(head);
            break;
        }
    } while (choice != 10);
}
```

```
---------menu-------
1.create lined list
 2.display
 3.insert_beg
 4.insert_end
 5.insert_before
 6.insert_after
 7.del_beg
 8.del_end
 9.del_node
 10.exitenter the choice:
1
Enter -1 to exit..
Enter the num: 40
Enter the num: 20
Enter the num: -1
linked list createdenter the choice:
2
40 20
enter the choice:
3
Enter the num: 10
enter the choice:
2
10 40 20
enter the choice:
4
Enter the num: 30
enter the choice:
2
10 40 20 30
enter the choice:
7
enter the choice:
2
40 20 30
enter the choice:
8
enter the choice:
2
40 20
```

```
Enter the num: 20
Enter the num: -1
linked list createdenter the choice:
2
40 20
enter the choice:
3
Enter the num: 10
enter the choice:
2
10 40 20
enter the choice:
4
Enter the num: 30
enter the choice:
2
10 40 20 30
enter the choice:
7
enter the choice:
2
40 20 30
enter the choice:
8
enter the choice:
2
40 20
enter the choice:
9
Enter the value that has to be deleted: 40
enter the choice:
2
20
enter the choice:
10

Process returned 0 (0x0)   execution time : 100.379 s
Press any key to continue.
```