**Data Structures**

**Prepared by Dr. Mohammed Al-Obaidi and Hussain M. Turki**

# Lecture 4
**Implementation and Construction of Singly Linked Lists in C++**

## Programs related to single linked list

### 1. Creation of a linked list

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct link
{
  int info;
  struct link *next;
};

struct link start;

/* function prototypes */
void create(struct link *);
void display (struct link *);
int main()
{
  struct link start, *node;
  create(node);
  display(node);
}
void create(struct link *node)
{
  char ch='y';
  start.next = NULL;
  node = &start; /* point to the start of the list */
  while(ch =='y' || ch=='Y')
  {
     node->next = new link();/*allocate a space in memory of
     size link*/
     node = node->next;
     cout<<"\n enter a number : ";
     cin>>node->info;
     node->next = NULL;
     cout<<"\n do you want to create more nodes: ";
     cin>>ch;
  }
}

void display(struct link *node)
{
  node = start.next;
  cout<<"\n after inserting a node list is as follows:\n";
  while (node)
  {
     cout<<"    "<<node->info;
     node = node->next;
  }
}
```

**2. insert a node into a simple linked list at the beginning**

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct list
{
  int info;
  struct list *next;
};

struct list start,*first, *newnode;

/*function prototypes*/
void create(struct list *);
void display(struct list *);
void insert(struct list *);

int main()
{
struct list start,*node;
create(node);
insert(node);
display(node);
}
void create(struct list *node)
{
  char ch='y';
  start.next = NULL;
  node = &start; /* point to the start of the list */
  while(ch =='y' || ch=='Y')
    {
        node->next = new list();/*allocate a space in memory of
        size list*/
        node = node->next;
        cout<<"\n enter a number : ";
        cin>>node->info;
        node->next = NULL;
        cout<<"\n do you want to create more nodes: ";
        cin>>ch;
    }
}
void display(struct list *node)
{
  node = start.next;
  cout<<"\n after inserting a node list is as follows:\n";
  while (node)
  {
     cout<<"  "<<node->info;
     node = node->next;
  }
}
void insert(struct list *node)
{ /*insert an element at the first node*/
  node = start.next;
  first = &start;
  newnode= new list();/*allocate a space in memory of size list*/
  newnode->next = node ;
  first->next = newnode;
  cout<<"\n input the first node value: ";
  cin>>newnode->info;
}
```

**3. insert a node into a simple linked list at the end of the list**

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct list
{
  int info;
  struct list *next;
};
struct list start, *first, *newnode,*last;

/* function main */
void create(struct list *);
void display (struct list *);
void insert(struct list *);
int main()
{
  struct list *node;
  create(node);
  insert(node);
  display(node);
}
void create(struct list *node) /*logic to create a link list*/
{
  char ch='y';
  start.next = NULL;
  node = &start; /* point to the start of the list */
  while(ch =='y' || ch=='Y')
  {
     node->next = new list();
     node = node->next;
     cout<<"\n enter a number : ";
     cin>>node->info;
     node->next = NULL;
     cout<<"\n do you want to create more nodes: ";
     cin>>ch;
  }
}
void display(struct list *node)
{ /*display the linked list*/
  node = start.next;
  cout<<"\n after inserting a node list is as follows:\n";
  while (node)
  {
     cout<<"    "<<node->info;
     node = node->next;
  }
}
void insert(struct list *node)
{ /* logic of insertion(last node) */
  node = start.next;
  last = &start;
  while(node)
  {
     node = node->next;
     last= last->next;
  }
  if(node == NULL)
  {
     newnode= new struct list();
```

```
        newnode->next = node ;
        last->next = newnode;
        cout<<"\n enter the value of last node: ";
        cin>>newnode->info;
   }
}
```

# Data Structure Lab

**Lecture ٤**

## Programs related to single linked list

### 1) Creation of a linked list

```
#include <iostream>
using namespace std;

struct link {
   int info;
   struct link *next;
};

link *start = nullptr; // جعل start مؤشرًا وإعداده كـ nullptr

// دالة لإنشاء القائمة المرتبطة
void create(link *&head) { // تمرير العنوان بالمرجع لضمان تحديثه في `main()`
   char ch = 'y';
   link *node, *temp;

   head = nullptr; // التأكد من أن القائمة فارغة في البداية

   while (ch == 'y' || ch == 'Y') {
      node = new link(); // إنشاء عقدة جديدة ديناميكيًا
      cout << "\nEnter a number: ";
      cin >> node->info;
      node->next = nullptr;

      if (!head) { // إذا كانت القائمة فارغة، اجعل العقدة الأولى هي `head`
         head = node;
      } else {
         temp->next = node; // ربط العقدة السابقة بالعقدة الجديدة
      }
      temp = node; // تحديث `temp` ليشير إلى العقدة الأخيرة

      cout << "\nDo you want to create more nodes? ";
```

```
        cin >> ch;
    }
}

// دالة لطباعة القائمة المرتبطة
void display(link *head) {
    cout << "\nThe list is:\n";
    while (head) {
        cout << " " << head->info;
        head = head->next;
    }
    cout << endl;
}

// دالة لتحرير الذاكرة ومنع تسربها
void freeMemory(link *&head) {
    link *temp;
    while (head) {
        temp = head;
        head = head->next;
        delete temp; // تحرير الذاكرة
    }
}

int main() {
    link *head = nullptr;

    create(head); // إنشاء القائمة
    display(head); // طباعة القائمة

    freeMemory(head); // تحرير الذاكرة قبل الخروج
    return 0;
}
```