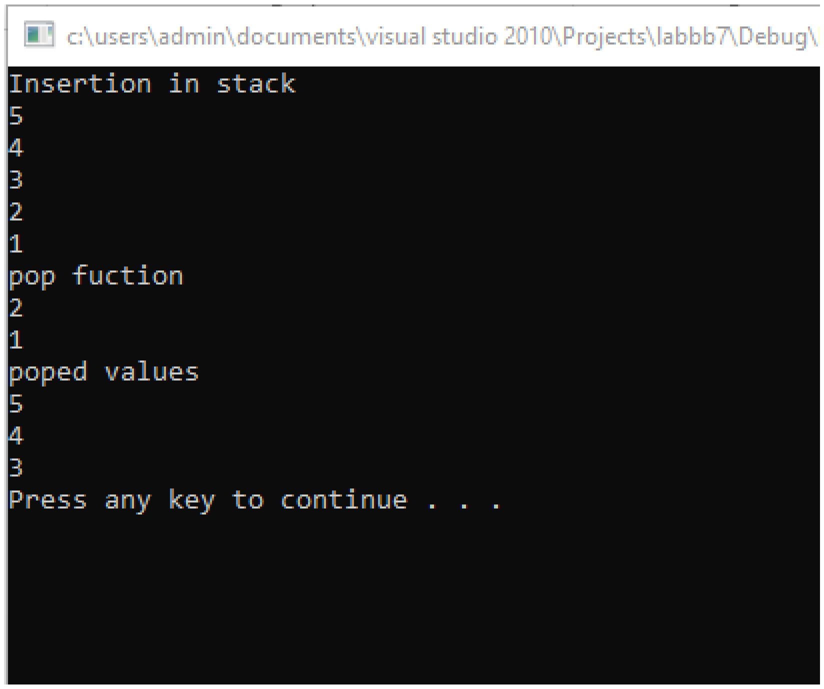
Exercise 1

A stack can be implemented using a linked list. The first node can serve as the 'top' of Stack and 'push' and 'pop' operations can be implemented by adding and removing nodes at the head of the linked list. Implement the Stack class using a linked list and provide all the standard member functions.

```
// labbb7.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include<iostream>
using namespace std;



class stack
{
private:
struct node
{int data;
node *next;
} *top;
public:
stack()
{top=NULL;
}
bool emptyList()
{if(top==NULL)
return true;
else
return false;}

int pop()
{
node **temp;
if(top==NULL)
{cout<<"linklist is empty"<<endl;
return 0;
}
else
       {node *temp=top;
top=temp->next;
temp->next=NULL;
int x;
x=temp->data;
delete temp;
return x;

}}
void push(int value)
{node*ptr=new node;
ptr->data=value;
ptr->next=NULL;
ptr->next=top;
top=ptr;
}

void traverse()
{node *ptr=top;
while (ptr!=NULL)
{cout<<ptr->data<<endl;
ptr=ptr->next;
}
}
};
int _tmain(int argc, _TCHAR* argv[])
{
       stack l;
cout<<"Insertion in stack"<<endl;
l.push(1);
l.push(2);
l.push(3);
l.push(4);
l.push(5);
l.traverse();
cout<<"pop fuction"<<endl;
int a,b,c;
a=l.pop();
b=l.pop();
c=l.pop();
l.traverse();
cout<<"poped values"<<endl;
cout<<a<<endl;
cout<<b<<endl;;
cout<<c<<endl;
       system("pause");
       return 0;
}
```



```
c:\users\admin\documents\visual studio 2010\Projects\labbb7\Debug\
Insertion in stack
5
4
3
2
1
pop fuction
2
1
poped values
5
4
3
Press any key to continue . . .
```

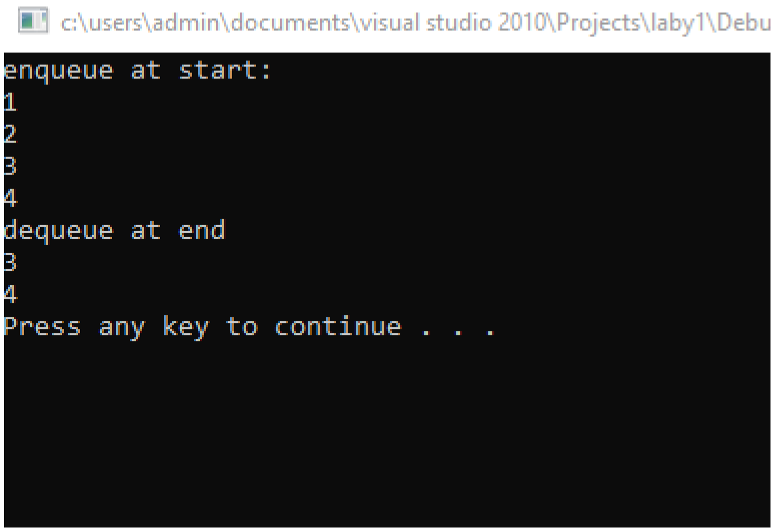
Exercise 2

Implement simple (Linear) Queue through linked list.

```
// laby1.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include<iostream>
using namespace std;


class queue
{
private:
struct node
{int data;
node *next;
} *front,*rear;
public:
queue()
{front=NULL;
rear=NULL;}
bool emptyList()
{if(front==NULL)
```

```cpp
return true;
else
return false;}

 void dequeue()
    {
        if (front == NULL)
        {
            cout << "Queue is empty" << endl;
        }
        else
        {
            node* temp = front;
            front = front->next;
            delete temp;

            if (front == NULL)
            {
                rear = NULL;
            }
        }
    }

    void enqueue(int value)
    {
        node* temp = new node;
        temp->data = value;
        temp->next = NULL;

        if (front == NULL)
        {
            front = temp;
            rear = temp;
        }
        else
        {
            rear->next = temp;
            rear = temp;
        }
    }

void traverse()
{node *ptr=front;
while (ptr!=NULL)
{cout<<ptr->data<<endl;
ptr=ptr->next;
}
}
};
int _tmain(int argc, _TCHAR* argv[])
{queue l;
cout<<"enqueue at start:"<<endl;
l.enqueue(1);
l.enqueue(2);
l.enqueue(3);
l.enqueue(4);
l.traverse();
cout<<"dequeue at end"<<endl;
```

```
l.dequeue();
l.dequeue();
l.traverse();

       system("pause");
       return 0;
}
```


c:\users\admin\documents\visual studio 2010\Projects\laby1\Debu

```
enqueue at start:
1
2
3
4
dequeue at end
3
4
Press any key to continue . . .
```

Exercise 3

Write the following C++ functions to realize the indicated functionality on a singly linked list of integers.

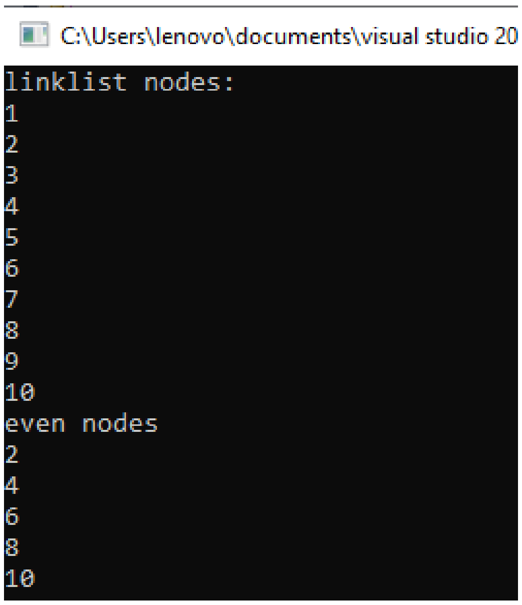
a. A function that prints only the even-numbered nodes of the list.

b. A function which takes two data values as arguments and swaps the node values containing them.

c. A function that deletes the first half of the linked list nodes. You are required to first determine the total number of linked list nodes and then delete the first part of the linked list.

```
// lab 7.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include<iostream>
```

```
using namespace std;


class List
{
private:
struct node
{int data;
node *next;
} *head;
public:
List()
{head=NULL;
}

bool emptyList()
{if(head==NULL)
return true;
else
       return false;}

void even_node()
{node*temp=head->next;
while(temp!=NULL)
{cout<<temp->data;
cout<<endl;
      temp=temp->next->next;
}}
void insert_begin(int value)
{node*ptr=new node;
ptr->data=value;
ptr->next=NULL;
ptr->next=head;
head=ptr;

}
void insert_end(int value)
{
    node* temp = new node;
    temp->data = value;
    temp->next = NULL;

    if (head == NULL) {
        head = temp;
    } else {
        node* s = head;
        while (s->next != NULL) {
            s = s->next;
        }
        s->next = temp;
    }
}

void traverse()
{node *ptr=head;
       while (ptr!=NULL)
       {cout<<ptr->data<<endl;
              ptr=ptr->next;
       }
}

};



int _tmain(int argc, _TCHAR* argv[])
{
       List l;
cout<<"linklist nodes:"<<endl;
l.insert_begin(5);
l.insert_begin(4);
l.insert_begin(3);
l.insert_begin(2);
l.insert_begin(1);
l.insert_end(6);
l.insert_end(7);
l.insert_end(8);
l.insert_end(9);
l.insert_end(10);
l.traverse();
cout<<"even nodes"<<endl;
l.even_node();
l.even_node();


       system("pause");

       return 0;
}
```



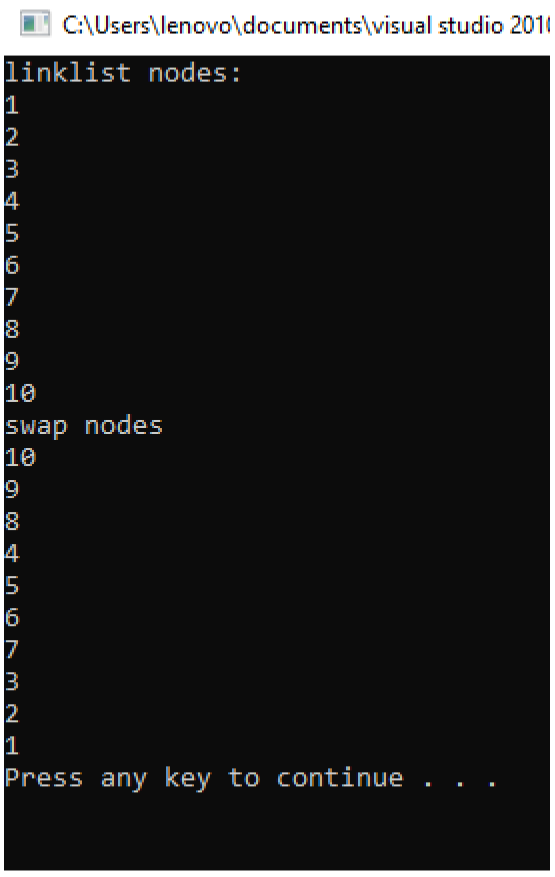
C:\Users\lenovo\documents\visual studio 20

```
linklist nodes:
1
2
3
4
5
6
7
8
9
10
even nodes
2
4
6
8
10
```

(B)

```
// lab 7.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include<iostream>
using namespace std;


class List
{
private:
struct node
{int data;
node *next;
} *head;
public:
List()
{head=NULL;
}

bool emptyList()
{if(head==NULL)
return true;
else
       return false;}
void swap(int a, int b)
{
    if (a == b) {
        // No need to swap if a and b are the same.
        return; }

    node* t1 = head;
    node* t2 = head;
    node* temp1 = NULL;
    node* temp2 =NULL;

    while (t1 != NULL && t1->data != a) {
        temp1 = t1;
        t1 = t1->next; }
 while (t2 != NULL && t2->data != b) {
        temp2 = t2;
        t2 = t2->next;}
if (t1 == NULL || t2 == NULL) {
        cout<<"null";}
if (temp1 != nullptr) {
        temp1->next = t2; }
else {
        head = t2;}
if (temp2 != nullptr) {
        temp2->next = t1;}
else {
        head = t1;}

    node* temp = t1->next;
    t1->next = t2->next;
    t2->next = temp;
}



void insert_begin(int value)
{node*ptr=new node;
ptr->data=value;
ptr->next=NULL;
ptr->next=head;
head=ptr;

}
void insert_end(int value)
{
    node* temp = new node;
    temp->data = value;
    temp->next = NULL;

    if (head == NULL) {
        head = temp;
    } else {
        node* s = head;
        while (s->next != NULL) {
            s = s->next;
        }
        s->next = temp;
    }
}

void traverse()
{node *ptr=head;
       while (ptr!=NULL)
       {cout<<ptr->data<<endl;
              ptr=ptr->next;
       }
}

};



int _tmain(int argc, _TCHAR* argv[])
{
       List l;
cout<<"linklist nodes:"<<endl;
l.insert_begin(5);
l.insert_begin(4);
```

```
l.insert_begin(3);
l.insert_begin(2);
l.insert_begin(1);
l.insert_end(6);
l.insert_end(7);
l.insert_end(8);
l.insert_end(9);
l.insert_end(10);
l.traverse();
cout<<"swap nodes"<<endl;
l.swap(1,10);
l.swap(3,8);
l.swap(2,9);
l.traverse();


       system("pause");

       return 0;
}
```

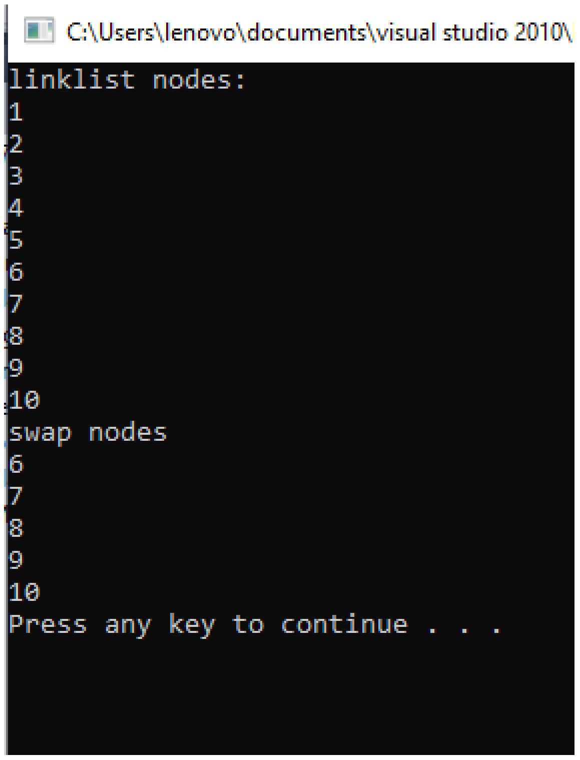


```
linklist nodes:
1
2
3
4
5
6
7
8
9
10
swap nodes
10
9
8
4
5
6
7
3
2
1
Press any key to continue . . .
```

(C)

```
// lab 7.cpp : Defines the entry point for the console application.
//
```

```
#include "stdafx.h"
#include<iostream>
using namespace std;


class List
{
private:
struct node
{int data;
node *next;
} *head;
public:
List()
{head=NULL;
}

bool emptyList()
{if(head==NULL)
return true;
else
       return false;}
void delete_half()
{int count=0;
node*temp=head;
while (temp!=NULL)
{count ++;
temp=temp->next;}
int half=0;
half=count/2;
for(int i=0;i<half;i++)
{node*ptr=head;
head=head->next;
ptr->next =NULL;
delete ptr;}

}



void insert_begin(int value)
{node*ptr=new node;
ptr->data=value;
ptr->next=NULL;
ptr->next=head;
head=ptr;

}
void insert_end(int value)
{
    node* temp = new node;
    temp->data = value;
    temp->next = NULL;

    if (head == NULL) {
        head = temp;
    } else {
        node* s = head;
        while (s->next != NULL) {
            s = s->next;
        }
        s->next = temp;
    }
}

void traverse()
{node *ptr=head;
       while (ptr!=NULL)
       {cout<<ptr->data<<endl;
              ptr=ptr->next;
       }
}

};



int _tmain(int argc, _TCHAR* argv[])
{
       List l;
cout<<"linklist nodes:"<<endl;
l.insert_begin(5);
l.insert_begin(4);
l.insert_begin(3);
l.insert_begin(2);
l.insert_begin(1);
l.insert_end(6);
l.insert_end(7);
l.insert_end(8);
l.insert_end(9);
l.insert_end(10);
l.traverse();
cout<<"swap nodes"<<endl;
l.delete_half();
l.traverse();


       system("pause");

       return 0;
}
```



C:\Users\lenovo\documents\visual studio 2010\

```
linklist nodes:
1
2
3
4
5
6
7
8
9
10
swap nodes
6
7
8
9
10
Press any key to continue . . .
```