Mid term exam   31/10/2007

اسم الطالب:.......................................   اسم المدرس:......................................

Student . #:_ _   Section #:_____   Seat #:_

Q1) Given the following definition of a **sllist**, use the functions whose prototype appear in the public section (if needed) to answer questions **A-C** :

class sllist
{
private:
        node * head; //pointer that points to the first node in a list
public:
        int delete_head( );   //deletes the first node
        void delete_node(int el);
};

A. Write a proper destructor for the above class.

   sllist::~sllist()   ➔1 Mark
   {
      while(head!= NULL) ➔1 Mark
              delete_head();   ➔1 Mark
   }

B. Add a definition of function delete_tail to a class sllist that will delete the last node from the above defined linked list and return the value of the deleted node.

   int sllist::delete_tail()   ➔1 Mark
   {
      int el;
      if (head->next == NULL)   ➔1 Mark
      {
              el = head ->info;
              delete head;   ➔1 Mark
              head = NULL;
      }
      else
      {
              node * tmp=head;   ➔1 Mark
              while(tmp->next->next != NULL) ➔1 Mark
                      tmp=tmp->next;
              el = tmp->next->info;
              delete tmp->next;   ➔1 Mark
              tmp->next=NULL;
      }
      return el;
   }

Instructor:Nabeel ALassaf

C. Write a definition of a function delete_odds that will delete all nodes that contain odd integers.

```
void sllist::delete_odds()
{
        node *tmp1=head,*tmp2=head->next;      ➔1 Mark
        while (tmp1!=NULL )        ➔1 Mark
        {
                if ((tmp1->info % 2) != 0)     ➔1 Mark
                {
                        cout<<"deleting "<<tmp1->info;
                        delete_node(tmp1->info);    ➔1 Mark
                }
                if(tmp2!= NULL)
                {
                        tmp1=tmp2;   ➔1 Mark
                        tmp2=tmp2->next;
                }
                else
                        tmp1=tmp2;

        }

}
```

Q2) Given the following definition of a **dllist,** use the functions whose prototype appear in the public section (if needed) to answer questions A,B :

```
template <class T>
class dllist
{
private:
        node<T> * current;  // pointer that points to any node in a list
public:
        void add_head(T  el);              // adds element at begin
        void add_tail (T el);              // adds element at end
        void add_at_pos(int pos,T el);        //adds element at position
};
```

A. Add a definition of a function print_back to class dllist that will print all elements of a list backwards**.**

```
Template <class T>          ➔1 Mark
Void dllist<T>::print_back( )
{
While(current->next!=NULL) ➔1 Mark
current=current->next;

While(current!=NULL) ➔1 Mark
{
Cout<< current->info;      ➔1 Mark
current=current->prev;     ➔1 Mark

}
}
```

B. Add a definition of a function add_sorted to a dllist that will add an element in a correct position into a liked list that contains elements sorted increasingly.

Template <class T>
Void dllist<T>::add_sorted(T el)
{
for (;current->prev!=NULL ;current=current->prev);➔1 Mark
int p=0; ➔1 Mark
while(current!=NULL && current->info<el) ➔2 Mark
{
current=current->next;
p++; ➔1 Mark
}
Add_at_pos(p,el);   ➔1 Mark
}

Q3) Count number of assignments in the following code, then find its complexity.

```
for(i=1 ; i<=m ; i+=2)
        for(j=i-2 ; j<=i+2 ; j++)
                sum+=a[j]
```

number of assignments is:

$=1+m/2+m/2+ 2* m/2 * 5$ ➔**3 Mark**
$=1+m+5m$
$=1+6m$
➔$O(m)$

Q4) Find complexity of the following code.

```
for( k=0 , j=1 ; j<=B ; j*=2)
        k++;
for ( j=1; j<=B ; j++)
        s++;
```

$=2+2*logB+1+2B$
$=3+2B+2*logB$
➔$O(B)$ ➔**2 Mark**

3