1. [30 pts] Briefly explain 5 of the following:

a)	friend class	d)	class hierarchy	g)	encapsulation	j)	linked list
b)	polymorphism	e)	virtual function	h)	stack vs. queue	k)	doubly-linked list
c)	inheritance	f)	information hiding	i)	is-a relationship	~~l)~~	~~class hierarchy~~

2. [40 pts]For this problem write **the class specification only** – do not write the implementation code for any class methods. Objects of the class(es) you write will eventually be used in a program to list a report of student grades and calculate overall class average, maximum, minimum, etc. but you are not working on that right now. Every student in CIS 4100 should have two exam grades and an average grade. You will need to code class specifications for the following:

- a Student class that has (at least) 2 data members: name, a string object, and gpa, a float object.
- a class UndergradStudent, derived from the Student class of part (a), that has (at least) data members credits, an integer, and major, a string object.
- a class CIS4100Student, derived from the UndergradStudent class of part (b), with data members: grade, a character object, the rest of the required attributes. You will also need the following methods: a function to compute the average and a function that uses the average to determine the students letter grade for the course based on the average.

Don't forget constructor, destructor, access, and modifier methods.

```
class Student{
public:
        Student(); ~Student();
        void setname(string); void setgpa (float);
        string getname(); float getgpa();
protected:
        string name; float gpa;
};

class UGStudent : public Student {
public:
        UGStudent(); ~UGStudent();
        void setcredits(int); void setmajor(string);
        int getcredits(); string getmajor();
protected:
        int credits; string major;
};

class CIS4100Student : public UGStudent {
public:
        CIS4100Student(int x, int y) {ex1=x; ex2=y; computeAvg();}
        void computeAvg() {avg=(ex1+ex2)/2.0;}
        char calcgrade(){/*switch or nested if for letter grade*/ }
        void printdata();
        ~CIS4100Student();
private:
        int ex1, ex2; float avg; char grade;
};

void  main(){
        //TBD
}
```

3. [30 pts]The following is a draft of a C++ program to process a linked list. Write the code for the missing **searchlist()** function.

```
//linkedlist.cpp  list is in no particular order; items are added at the front
#include <iostream>
using namespace std;
class node{ //one node in a linked list
public:
       node(){};
       ~node(){};
```

```
        int getdata(){return data;} //access functions
        node* getnextPtr() {return  nextPtr;}
        void setdata(int k){data=k;} //modifier functions
        void setnextPtr(node *p){nextPtr=p;}
private:
        int data;
        node* nextPtr;
};
class list{//a linked list
public:
        list(){node* firstPtr = NULL;} // creates a new empty list
        ~list(){};
        void EnterListData();
        int searchlist(int);
private:
        node *firstPtr;
};
void list::EnterListData(){
int val;
        node* p;
        cin >> val; //read and add values to the list. A negative number ends the input
        while (val >= 0) {
                p = new node;
                p->setdata(val);
                p->setnextPtr(firstPtr);
                firstPtr = p;
                cin >> val; }
}
int main(){
        list MyList;
        MyList.EnterListData();
        int val;
        cout << "\nEnter target: "; cin >> val;
        if (MyList.searchlist(val))
                cout << "target found" << endl;
        else cout << "target not found" <<  endl;
        return 0;
}
```

```
int list::searchlist(int t){
        int found=0;
        node *p = firstPtr;
        while (p != NULL) {
                if (p->getdata() == t) found=1;
                p = p->getnextPtr();
        }
        return found;
}
```