CSCI 131-- Techniques of Programming

College of the Holy Cross

Solutions to Final exam Review problems

The following problems are intended to help you study for the exam. Note that there will also be questions covering topics from the first three midterms, so you should be sure to review those topics as well. Use your Exams and Review sheets for those exams to review.

1) Write a function that counts how many even numbers there are in an integer linked list. The function should have one parameter, a pointer to a linked list. The function should not alter the list. It should return an integer value equal to the number of even numbers in the list.

You can assume the following definitions:

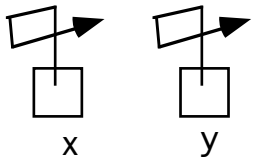
```
struct Node {
        int data;
        Node *next;
}
typedef Node *NodePtr;


int CountEvens( NodePtr theList ) {
        int count = 0;
        NodePtr currNodePtr = theList;
        while ( currNodePtr != NULL ) {
                if ( ( currNodePtr - > data % 2) == 0 ) {
                        count ++;
                }
                currNodePtr = currNodePtr - > next;
        }
        return count;
}
```

2) Consider the following code:

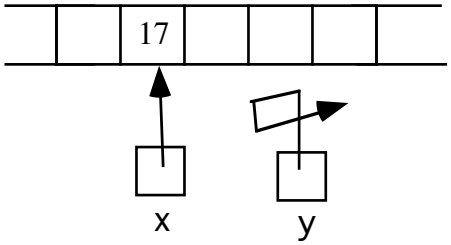
```
int *y;
int *x;
x = new int;
*x = 15;
*x += 2;
y = x;
*y *= 2;
```

a) Draw a diagram to show represent the pointers as each line of this code is executed.

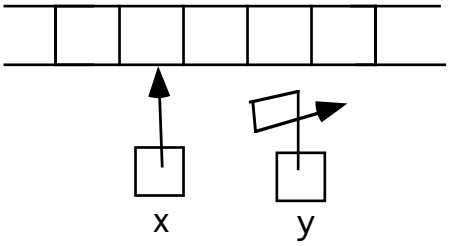
Lines 1 and 2:



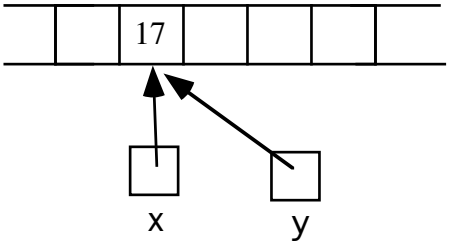
Line 5:



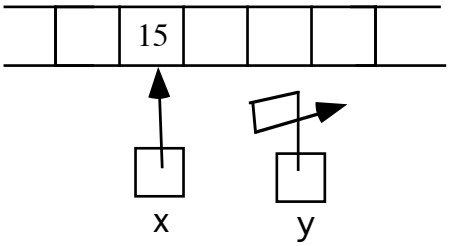
Line 3:



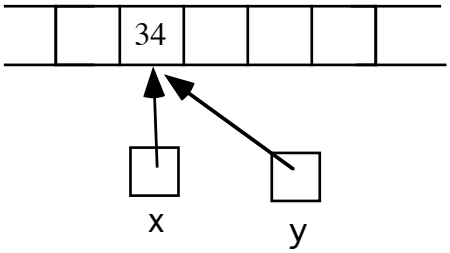
Line 6:



Line 4:



Line 7:



b) What are the values of *x and *y after executing this code?

Answer: Both have value 34

3) Consider the following recursive function:

```
int Mystery( char myString[ ], char theChar, int first, int last) {
        int answer = 0;
        if (first > last) {
                return 0;
        } else {
                answer = Mystery ( myString, theChar, first + 1, last);
                if ( myString[first] == theChar) {
                        return answer + 1;
                } else {
                        return answer;
                }
        }
}
```

a) Indicate which lines of code represent the solution to the base case. Also state what the base case is.

The Base case occurs when first > last.
Solution to the base case:

```
return 0;
```

b) Indicate which lines of code represent the solution to the general case.

```
answer = Mystery ( myString, theChar, first + 1, last);
if ( myString[first] == theChar) {
        return answer + 1;
} else {
        return answer;
}
```

c) Suppose you have the following code use to call the Mystery function:

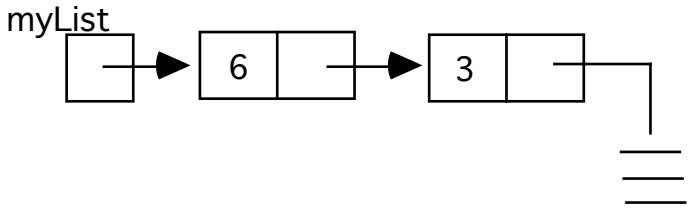
```
char theString = "CSCI";
int test;
test = Mystery(theString, 'C', 0, 4);
```

Draw a trace of the values of the following parameters as each recursive function call is executed:

first	last	myString[first]	answer	return value
0	4	'C'	1	2
1	4	'S'	1	1
2	4	'C'	0	1
3	4	'I'	0	0
4	4	'\0'	0	0
5	4	---	---	0

d) What does the Mystery function do?
Answer: It counts the number of time theChar occurs in myString between the first and last indices.

4) Consider the following linked list:



Assume a Node is represented by the following struct:

```
struct Node {
        int  myNumber;
        Node *nextNode;
};
typedef Node *NodePtr;
```

a) Write an expression to refer to the integer in the second node of the list.

```
myList - > nextNode - > myNumber
```

b) What is the value of the following expression?

```
myList - > nextNode - > nextNode
```

NULL

c) Write the C++ code to create and insert a new node, whose data value is 4, in between the two nodes of myList.

```
NodePtr newNodePtr;
newNodePtr = new Node;
newNodePtr - > myNumber = 4;
newNodePtr - > nextNode = myList - > nextNode;
myList - > nextNode = newNodePtr;
```

5. Consider the following class declarations:

```
class Student {
        public:
                void SetName( char name[ ]);
                void SetID( int idNum);
                void Write( ) const;
                void SetGpa(float the_GPA);
                Student ( );
                Student (char initName[ ], int initId);
        private:
                char name [25];
                int id;
                float gpa;
};

class StudentYear : public Student {
        public:
                void SetYear( int year);
                void Write ( ) const;
                void incrementYear( );
                StudentYear ( );
                StudentYear (char initName[ ], int initId, int initYear);
        private:
                int year;
};
```

a) Which class is the parent class and which is the child class?

Parent (or base) class: Student
Child (or derived) class: StudentYear

b) Which function is the same for both the Student and the StudentYear class?

SetGpa(float gpa), SetID(int idNum) and SetName(char name[ ]) are the same for both. They are inherited from Student by StudentYear.

c) What are the private data members for each class?

Student: name[ ], id, gpa
StudentYear: year, name[ ], id, gpa

d) Which private data members can be accessed directly by the member functions of each class?

Student: name[ ], id, gpa
StudentYear: year

e) What specific kind of function is StudentYear( )?
It is a constructor function.

f) Write the implementation of the Write( ) function for the Student class. This function should write to the standard output the values of name, id and gpa of the Student object.

```
void Student::Write( ) const {
        cout << "Student name: " << name << endl;
        cout << "Student ID: " << id << endl;
        cout << "Student GPA: " << gpa << endl;
}
```

g) Write the implementation of the Write( ) function for the StudentYear class. This function should write to the standard output the values of name, id, gpa and year of the StudentYear object.

```
void StudentYear::Write( ) const {
        Student::Write( );
        cout << "Graduation year: " << year << endl;
}
```

h) Write the implementation of the Student(char initName[ ], int initId) function for the student class. This function should assign initName[ ] to the name data member and initID to the id data member of the Student object.

```
Student::Student(char initName[ ], int initId) {
        strcpy(name, initName);
        id = initID;
}
```