C++ Programming Problems

C++ Problem

Problem C++1: Customer Class
Assume that a customer is characterized only by a name, address, and shoe size (as a floating point value). Write a corresponding Customer class to model the customer. Its properties should not be directly accessible and changeable. Implement a copy constructor, a setter for the address, and a getter for the shoe size. Use the const keyword correctly in all the previously mentioned components of the class.
You do NOT need to write a main function. You can assume that the data passed as parameters will be valid.

C++ Problem

Problem C++2: Vector Class with Dynamic Memory
Write a parametric constructor and a destructor for the class below. Consider the fact that memory has to be allocated and deallocated. You do NOT need to write a main function. You can assume that the data passed as parameters will be valid.

```
class vector {
private:
    int *v;
    int size;
public:
    // declaration
};
// add definitions
```

C Programming Problems - Files

C Problem

Problem C1: Distance Conversion
Write a program fragment which opens a file called "distance.txt" which contains two values per line reflecting a distance value represented in yards and feet:

```
1 1
1.4 2
2.6 8.7
1 2.8
```

Your program fragment should read the content of the file and generate another file called "distances_meters.txt" which contains per line a line container and the converted distances to meters separated by space. Consider that 1 yard = 0.9144 meters and 1 foot = 0.3048 meters. The writing should be done using fwrite. Do not forget to close the file.

C Problem

Problem C2: Grade Average Calculation
Write a program fragment which opens a file called "input_grades.txt" which has the following structure (matriculation number, grade, grade) like in the example below:

```
3000101 55 75
3000124 90 85
3000231 100 98
3000424 60 70
```

Your program fragment should read the content of the file and generate another file called "average.txt" which contains per line matriculation number, space, average grade (of the two grades) and newline. The writing should be done using fwrite. Do not forget to close the files.

Bitwise Operations Problems

Bitwise Problem

Problem B1: Check Binary Ending
Write a function:

```
int endswith111(unsigned char x);
```

which receives an unsigned character.
The function should return 1 if the binary representation of x ends with 111 and 0 if it does not end with 111 (positions 2, 1, and 0 considering from right to left). This function should test the binary representation of x for its ending using bitwise operations (i.e., operations on the number at the level of the decimal representation are not allowed).
You should not use any arrays, and you should not convert to binary explicitly.

Bitwise Problem

Problem B2: Count 1's in Binary
Write a function:

```
int count1s(unsigned char x);
```

which receives an unsigned character. The function should return the number of 1's in the binary representation of x. This function should test the binary representation of x for the amount of 1's using bitwise operations (i.e., operations on the number at the level of the decimal representation are not allowed).
You should not use any arrays and you should not convert to binary explicitly.

Linked List Problems

Linked List Problem

Problem L1: Circular Linked List Insertion
Assume that you have a main function that reads integer values from the standard input and performs adding with respect to a circular singly linked list. You DO NOT have to write this main function.
Write a function `void add(int x, struct circlist *list)` that implements the insertion of the value x after the "last" timely added node into the circular list.
Use the two following data structures:

```
struct node {
    int value;
    struct node *next;
};

struct circlist {
    struct node *start;
};
```

Originally, the circular list is empty.
For example, if the circular list has the following content: start $\rightarrow$ 1 $\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ (back to start)
Then after adding the value 4, the list will look like: start $\rightarrow$ 1 $\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ 4 $\rightarrow$ (back to start)

Linked List Problem

Problem L2: Even/Odd Linked List Separation
Assume that you have a main function that reads integer values from the standard input and adds them to an array of two linked lists. One list is for odd numbers and the other for even numbers. The lists are members of an array of length 2. You do NOT have to write this main function.
Write a function `void add_end(struct node **root, int number)` that implements the insertion of a number at the end of one of the two lists depending on the fact that it is odd or even.
Use the two following data structures:

```
struct node {
    int value;
    struct node *next;
};

struct node *root[2];
// root[0] points to the list of odd numbers
// root[1] points to the list of even numbers
```

Matrix Problems

Matrix Problem

Problem M1: Print Upper Triangular Matrix
Complete the following program fragment such that it reads from the keyboard the elements of a matrix a of size n × n consisting of n rows and n columns. Your program should also print on the screen the elements on and above the main diagonal of the Matrix as shown in the following example.

```
int n;
scanf("%d", &n);
```

Sample input:

```
4
1 2 3 4
5 6 7 8
9 1 2 3
4 5 6 7
```

Sample output:

```
1 2 3 4
  6 7 8
    2 3
      7
```

You can assume that the input will be valid, that the entered matrix will have 100 columns and 100 rows, and that the matrix values will have only one digit.

String Problems

String Problem

Problem S1: Alternating Strings
Write the definition of the function with the following prototype:

```
char *alternate(char *S1, char *S2);
```

which returns a new dynamically allocated string built within the function containing the combination of S1 with S2 by alternating S1 and S2 character by character. After the shorter string is consumed during the alternation, the remaining of the other string should be copied at the end of the result. For example, if S1 = "onestring" and S2 = "another" then the result of the alternation is the new string returned by the function which is "oanneostthreirng". To simplify the definition of the function you can safely assume that S1 is longer than S2 as in the example.

String Problem

Problem S2: Replace Spaces with Word Length
Write the definition of the function with the following prototype:

```
char *replace_spaces(char *s);
```

which replaces the spaces within the string s by the length of the word before the space and adds the length of the last word to the end. You can assume that you have a function which returns a character corresponding to the length like `char nr2char(int length)`. You do NOT have to write this function. The function replace_spaces is responsible for the exact dynamic memory allocation and generation of the previously described new string.
For example, if s = "one and three and lastword" then the new string returned by the function will be result = "one3and3three5and3lastword8".
You can assume that the string parameter will be non-empty and valid in terms of content. For simplicity, you can assume that for the length of a word you need only one character.

Recursive Functions Problems

Recursive Problem

Problem R1: Print Even Numbers Forward
Write a recursive function with the following prototype:

```
void print_even_forward(int nr);
```

which prints forward on the screen the even positive integer numbers starting from 2 until nr separated by space without creating an array. For example, if nr is 14 then 2 4 6 8 10 12 14 should be printed. If nr is 11 then 2 4 6 8 10 should be printed. You can assume that nr is a non-zero positive integer.

Recursive Problem

Problem R2: Print Odd Numbers Backwards
Write a recursive function with the following prototype:

```
void print_odd_backwards(int n);
```

which prints backwards on the screen the odd positive integer numbers starting from n separated by space without creating an array. For example if n is 14 then 13 11 9 7 5 3 1 should be printed. You can assume that n will be a non zero positive integer.