December 2017/98-Comp-A4

Program Design and Data Structures

3 Hours Duration

Notes:

1. If doubt exists as to the interpretation of a question, the candidate is urged to submit with the answer paper a clear statement of any assumptions made.

2. No calculator permitted. This is a Closed book exam.

3. Answer any six of the nine questions.

4. Any six questions constitute a complete paper. Only the first six questions as they appear in your answer book will be marked.

5. For questions that ask the candidate to write a program, **pseudocode** or any high-level language (e.g. **C** or **C++**) is acceptable unless otherwise specified. In all cases, marking will emphasize the operation of the program and not syntactic details.

6. All questions have equal weight. The total mark is out of 120.

Question 1:	(a) 10 marks; (b) 10 marks.
Question 2:	20 marks.
Question 3:	20 marks.
Question 4:	20 marks.
Question 5:	20 marks.
Question 6:	20 marks.
Question 7:	(a) 10 marks; (b) 10 marks.
Question 8:	(a) 15 marks; (b) 5 marks.
Question 9:	20 marks.

Question 1. *Programming.*

(a) Floyd's triangle lists integers in a right triangle that is aligned to the left. The first row is simply the integer 1. Successive lines start with the next integer followed by successive integers listing one more integer than the line above. Here is what the triangle looks for 5 lines:

```
 1
 2  3
 4  5  6
 7  8  9 10
11 12 13 14 15
```

Write a program to read the number of rows `n` from the user and then generate and output the first `n` lines of Floyd's triangle.

(b) Three desperados stole a shipment of gold bars late one night. After escaping to their hideout, they resolved to divide the booty in the morning and went to bed. However, as soon as one of the bandits heard the others snoring, he split the stolen gold into three equal piles, and found one bar left over. He buried one of the three piles under a tree, along with the extra bar, and left the rest of the gold. Then he went to sleep, sure that he had protected his interest in the treasure. Naturally, the other two outlaws were no more honest than the first. Each in turn crept to the cache of gold, divided it three ways, and found one bar left over, which he kept along with "his third".

Soon came the morning and the final three-way division. Oddly enough, this division also left one bar remaining. The highwaymen fought over this bar, and in an unprecedented three-way draw, shot each other dead.

Write a program to find how many bars could have been in the original loot. Assume that the loot contained no more than 500 bars.

Hint: Write a subprogram to determine if the above divisions of the gold are possible for a given number of bars and then use it.

Question 2. *Programming.*

A sequence of numbers known as a *hailstone* series is formed as follows:

- A starting number is picked
- If it is odd, the number is tripled and one is added
- If it is even, the number is divided by two

Although the numbers in the sequence go up and down, they eventually they reach a "ground state" of 4 2 1 4 2 1…. This has been shown for every number up to about 1.2e12.

Write a program that prompts the user for a starting number. The program should answer the following two questions:

1. How many steps (counted by the number of times a number is tripled or is divided by two) does it take to get to the repeating ground state?
2. What is the largest number the sequence reaches along the way?

For example, if the user enters 10, it takes 6 steps to reach the ground state. The largest number reached is 16.

Question 3. *Programming.*

The *saddle element* in an array is the element that is simultaneously the smallest element in its row and the largest element in its column. For example, the element `A(3,3)=4` is the saddle element in the following 6x6 array:

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

Write a program to input an `nxn` two-dimensional array, determine the saddle element in the array, and print the value and location of the element. Assume that there is exactly one saddle point in the array. Also assume that all integers in the array are between 1 and 99.

Question 4. *File I/O.*

Write a program to determine if two input files are "highly similar", i.e., are identical ignoring any while spaces (i.e., blanks, tabs and new lines). The input file names are to be read by your program from the standard input. Your program should return true if the files are highly similar and false otherwise. The two files may or may not be of equal sizes. Your code should return false as soon as it determines that the input files are not highly similar.

Question 5. *File I/O.*

Write a program to *merge* two sorted input files into one sorted output file. Assume each file has a number of "records", each consisting of a sequence of characters (including white spaces) terminated by a newline character, \n (assume a maximum record size of 80 characters including the terminating newline). The input files may have different numbers of records. Prompt the user for the names of the three files.

It is best that you start your answer with a short paragraph that describes your strategy for solution, the follow with the code. Include comments in your code!

Question 6. *Object-Oriented Design.*

Matrices are used in many engineering applications. However, most languages, including **C++** and **Java**, do not have a "matrix" data type, nor do they directly support matrix operations.

Design and write a **C++** class (call it **`Matrix`**) for supporting matrices and their operations. Your class should allow for the declaration of a matrix of a given size (rows, columns), with and without initialization of its elements. Your class should allow for the accessing (read & write) of elements of a matrix. It should also allow for the addition, subtraction, multiplication, and printing of matrices.

Assume only matrices of type `double` for simplicity. Overload the usual arithmetic operators to provide addition, subtraction, and multiplication of two matrices. Also overload the equality operator to allow equality comparison of two matrices.

You have freedom to select the exact syntax of some of the above operations. State any assumption you make clearly. Separate your class into a **`Matrix.h`** header file and a **`Matrix.cc`** implementation file.

Question 7. *Pointer-based Data Structures.*

(a) An element of a doubly linked list can be defined as follows, expressed in C:

```
typedef struct element {
    int data;
    struct element *prev;
    struct element *next;
} ELEMENT;
```

Write a function `del_dupl()` that deletes duplicate valued elements in a doubly linked list. The header of the function is shown below. The function must work correctly for empty lists.

```
/* delete duplicate valued elements in the list pointed to
   by head
*/
void del_dupl(ELEMENT *head);
```

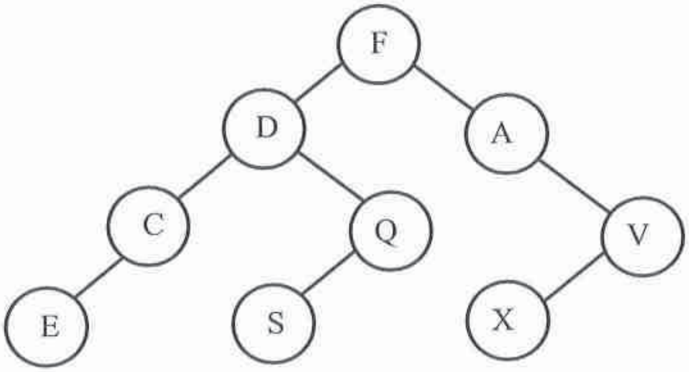
(b) Consider the following definitions of a `node` structure and of a `stack` module.

```
typedef struct {
    int data;
    node *next;
} node;


#ifndef STACK_H
#define STACK_H

static node *top = NULL;

void make_empty(void);
int is_empty(void);
void push(int i);
int pop(void);

#end if
```

Write an implementation of the `stack` module using a linked list of `nodes`. Recall that a stack is a list of elements with insertions and deletion done at only one end of the list, called the "top" of the stack.

Question 8. *Binary Trees.*

(a) Give the *inorder*, *preorder* and *postorder* traversals of the tree shown below.



(b) A tree ***T*** has the following inorder and preorder traversals:

```
Preorder traversal:      1 8 12 25 13  7 9
Inorder  traversal:      8 1 25 12  7 13 9
```

Draw **the** tree ***T***. Please note that there is only one tree ***T*** that has the inorder and preorder traversals shown above. Do **NOT** draw two trees; draw only one tree whose inorder and preorder traversal are as shown above.

Question 9. *Algorithm Design.*

A **palindrome** is a sequence of characters that reads the same forwards as backwards. For example, "bob", "deed" and "level" are palindromes, while "test" and "Hello" are not.

Write a **recursive** function: **`int palindrome (char *seq, int n)`** that returns 1 if the sequence of n characters pointed to be **`seq`** is a palindrome, and returns 0 otherwise.

Your solution should not exceed a few lines of code.