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.

Marking Scheme

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

Total mark is out of 120, which will then be normalized to 100.

Question 1. *Programming.*

(a) Write a program that prompts the user to input a birthdate and responds by printing the horoscope sign corresponding to the birthdate. The birthdate format is month (1-12), followed by a space, then followed by a day (1-31). The program should check for invalid months or invalid days within a month.

Here are some examples:

```
Enter birthdate: 10 18
Sign is Libra

Enter birthdate: 1 12
Sign is: Capricorn

Enter birthdate: 2 30
Invalid birthdate
```

Here are the horoscope signs and their dates:

Aries	March 21 – April 19
Taurus	April 20 – May 20
Gemini	May 21 – June 21
Cancer	June 22 – July 22
Leo	July 23 – August 22
Virgo	August 23 – September 22
Libra	September 23 – October 22
Scorpio	October 23 – November 21
Sagittarius	November 22 – December 21
Capricorn	December 22 – January 19
Aquarius	January 20 – February 18
Pisces	February 19 – March 20

(b) Horoscope signs of the same *Element* are most compatible. There are 4 Elements in astrology and 3 signs in each: FIRE (Aries, Leo and Sagittarius), EARTH (Taurus, Virgo and Capricorn), AIR (Gemini, Libra and Aquarius) and WATER (Cancer, Scorpio and Pisces). One is most compatible with a person with the same sign or the other two signs in the same Element.

Extend your program in part **(a)** to also print the other two signs a birthdate is most compatible with.

Question 2. *Programming.*

A problem that often arises in the design of Very Large Scale Integration (VLSI) circuits is to determine whether or not two circuit wires inadvertently intersect. Let's assume that for all practical purposes wires are either vertical or horizontal. A pair of endpoint coordinates $(x_1,y_1:x_2,y_2)$ describes each wire. Hence, (6,8:3,8) is a horizontal wire and (2,4:2,6) is a vertical one. Consequently, a simple check of the x and y coordinates of two wires can reveal if the two wires intersect.

Write a program that reads 20 pairs of coordinates and determines which wires intersect. Your program should print the coordinate pairs of intersecting wires.

Hint: read endpoint pairs into two arrays.

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. *Object-Oriented Design.*

Sets of numbers are used in many applications. However, some languages, like **C++**, do not have "Set" as a data type, nor do they directly support set operations.

Design and write a **C++** class (call it **Set**) for supporting sets and their operations. Your class should allow for the declaration of sets, both empty and initialized with elements. It should allow for the following set operations: addition of an element, deletion of an element, and checking if an element is a member of the set. Your implementation should allow for sets of various number types (i.e., sets of integers, sets of floats, etc).

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

Question 5. *File I/O.*

Write a program to generate personalized mail. The program takes input both from an input file and from the keyboard. The input file contains the text of a letter, except that the first name of the recipient is indicated by the three characters #N#. The program prompts the user for a name and then writes the letter to a second file, but with the three characters replaced by the name. The three characters #N# may occur more than once in the file.

Question 6. *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) A node in a binary tree can be defined as follows, expressed in C:

```
typedef struct treenode {
    int data;
    struct element *left;
    struct element *right;
} TreeNode;
```

Write a function `inorder()` that traverses the tree using inorder traversal. The header of the function is shown below. The function must work correctly for an empty tree. Assume at each node, data is printed to the standard output.

```
/* Inorder traversal
*/
void inorder (TreeNode *root);
```

Question 7. *Binary Trees.*

Consider the following sequence of binary tree nodes. The key of each node appears inside the node.

(a) Draw the binary search tree that results from the successive insertion of the nodes in the order in which they appear above from left to right.

(b) Delete the node with the key 14 and redraw the resulting binary search tree.

(c) Re-insert the node with the key 14 and re-draw the resulting binary search tree.

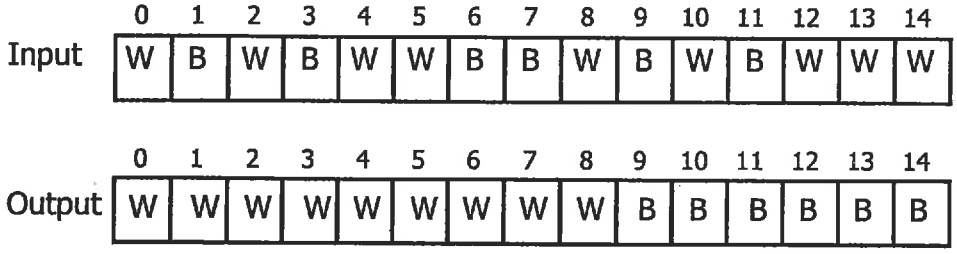
(d) Give the inorder, preorder, and postorder traversals of the tree in part (a) above.

Question 8. *Sorting.*

Insertion sort, Bubble sort, and Quicksort are three popular methods for sorting elements in an array of size n. Give the time complexity for each method as a function of n. Which of these methods would you use if you had to sort 30 elements? Which would you use if you had to sort 3 million elements? Justify your answer in each case.

Question 9. *Algorithm Design.*

Consider the following variation of the *Dutch National Flag* problem described by E.W. Dijkstra. Suppose an array of length N holds two different values: white and black. Write a program that puts all the white values to the left of the array and the black values to the right. Here is an example for a 15-element array:



Assume for simplicity that the array is of integer type, and that 0 represents white, while 1 represents black.

You may not use a sorting program, nor may you go through the array to count the number of 0's and 1's. That is, you must solve the problem in one pass or traversal of the array.

Your solution should not exceed a few lines of code.