

112年專門職業及技術人員高等考試建築師、
25類科技師（含第二次食品技師）、大地工程
技師考試分階段考試（第二階段考試）
暨普通考試不動產經紀人、記帳士考試試題

等 別：高等考試

類 科：資訊技師

科 目：程式設計

考試時間：2 小時

座號：_____

※注意：(一)禁止使用電子計算器。

(二)不必抄題，作答時請將試題題號及答案依照順序寫在試卷上，於本試題上作答者，不予計分。

(三)本科目除專門名詞或數理公式外，應使用本國文字作答。

一、使用 C++ 或 Java 設計一個函式 (function, method)，傳入一個大於或等於 2 的整數，可以列出輸入之整數的質因數乘積。(20 分)

注意：符號 $\wedge$ 代表次方。

執行範例：

foo(13)輸出 $13 = 1 * 13$

foo(30)輸出 $30 = 2 * 3 * 5$

foo(675)輸出 $675 = 3 \wedge 3 * 5 \wedge 2$

二、一個骰子遊戲的計分規則如下所述：一個骰子共有六面，點數分別為 1~6。遊戲時擲出三個骰子，若三個骰子的點數相同，則分數為三個骰子點數總和再乘以 3；若其中有兩個骰子點數相同，另一個不同，則分數為點數不同骰子的點數。若三個骰子的點數都不同，則重新再擲三個骰子，直到出現有分數的情況為止。請使用物件導向程式語言 C++ 或 Java 為此遊戲設計程式。(20 分)

執行範例一：

骰子點數：3 3 6; 得分：6

執行範例二：

骰子點數：2 2 2; 得分：18

執行範例三：

骰子點數：1 2 3; 重擲

骰子點數：6 5 4; 重擲

骰子點數：5 1 5; 得分：1

三、二元搜尋樹 (binary search tree) 是一種資料經過整理後的二元樹。對於任何一個節點，若左子樹 (left subtree) 存在，則左子樹所有節點的值均小於自己的值；若右子樹 (right subtree) 存在，則右子樹所有節點的值均大於自己的值。請用 C++ 或是 Java 寫一個二元搜尋樹的類別 (class)，包括新增一個節點 (insert)、刪除一個節點 (remove) 以及中序走訪 (inorder) 這三個函式 (function) 並撰寫主程式完成以下工作。(40 分)

工作項目：

- 依序新增 25, 15, 20, 30, 40 到一個空的二元搜尋樹。
- 以中序走訪此樹
- 刪除 30
- 增加 30
- 以中序走訪此樹

四、請寫出下列 Java 語言程式的執行結果，也就是 `System.out.println("Sum before the swap: " + Element.sum(e))`及 `System.out.println("Sum after the swap: " + Element.sum(e))`的輸出結果。(20 分)

// **Trace.java** (檔名)

```
public class Trace{
    public static void main(String[] args){
        Element[] e = new Element[5];
        e[0] = new Element(10);
        e[1] = new Element(5);
        e[2] = e[1];
        e[3] = e[0];
        e[4] = new Element(5);
        System.out.println("Sum before the swap: " + Element.sum(e));
        for(int i = 0; i < e.length - 1; i++){
            if(e[i] == e[i+1]){
                break;
            }
            else{
                Element eTmp = e[i];
                e[i] = e[i+1];
                e[i+1] = eTmp;
            }
        }
        System.out.println("Sum after the swap: " + Element.sum(e));
    }
}
```

// **Element.java** (檔名)

```
public class Element{
    private int value;

    public Element(int value){
        this.value = value;
    }
    public int getValue(){
        return value;
    }
    public static int sum(Element [] e){
        int sum = e[0].getValue();
        for(int i = 0; i < e.length - 1; i++){
            if(e[i].getValue() <= e[i+1].getValue()){
                continue;
            }
            sum += e[i+1].getValue();
        }
        return sum;
    }
}
```