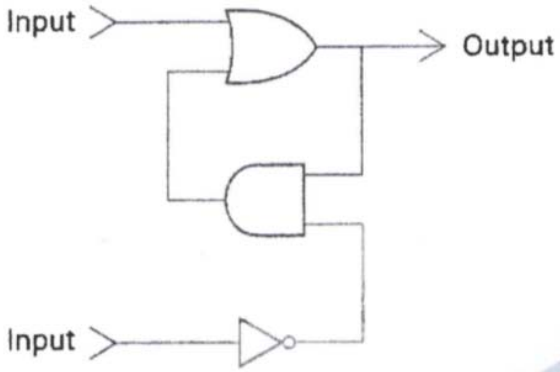
1. 請問以下 flip-flop 電路之兩輸入皆為 0 時，其輸出值為何？



(A) 0
(B) 1
(C) 未定義
(D) 保持上次的輸出值
(E) 以上皆非

【講義命中】計算機概論第二回 P.48~P.49，觀念完全命中，唯獨問法與數值不一樣

一、邏輯閘

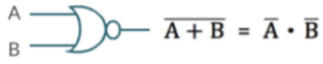
(一)AND gate

(二)OR gate

(三)NOT gate

(四)NAND gate
將AND輸出後再NOT

(五)NOR gate
將OR輸出後再NOT



(六)XOR gate（Exclusive-OR gate）

1. $A \oplus B = \bar{A}B + A\bar{B}$
2. 相同為0，相異為1。
3. 輸入資料有偶數個「1」，其結果必為「0」。
4. 輸入資料有奇數個「1」，其結果必為「1」。
5. $A \oplus 1 = \bar{A}$ ， $A \oplus 0 = A$

(七)XNOR gate（Exclusive-NOR gate）

1. $A \odot B = \bar{A}\bar{B} + AB$
2. 與XOR相反
3. 輸入資料有偶數個「0」，其結果必為「1」。
4. 輸入資料有奇數個「0」，其結果必為「0」。
5. $A \odot 1 = A$ ， $A \odot 0 = \bar{A}$

試證明：$\overline{A \oplus B} = A \odot B$

$$\overline{A \oplus B} = \overline{(\bar{A}B + A\bar{B})} = (A + \bar{B}) \cdot (\bar{A} + B) = A\bar{A} + AB + \bar{A}\bar{B} + B\bar{B}$$
$$= AB + \bar{A}\bar{B} = A \odot B$$

4. 以下作業系統之敘述何者錯誤？

(A) 批次處理（Batch Processing）為早期作業系統的設計之一，其目的是讓使用者與機器使用分離，使得使用者可以遞送工作（job）至主機分批執行。

(B) 互動式處理（Interactive Processing）使得使用者可以透過終端機（terminal）與程式互動。

(C) 分時系統（Time-sharing）可以提供多個使用者同時使用同一機器。

(D) Context Switch 機制可以提供多個程式分享使用同一個 CPU。

(E) 以上敘述皆正確。

【講義命中】計算機概論第五回 P.2~P.3，相似度 90%

(二)作業系統類型

1. 批次系統（Batch system）

早期電腦的主要工作即是從一個工作到下一個工作時的自動控制轉換，此種作業系統會常駐於記憶體中。在使用此種系統時，為了提升效率，會將類似的工作集中，並於同一時間交給電腦處理。

2. 多重程式（Multiprogramming）

此種系統的主要目的為讓CPU一直保持在忙碌的狀態，來提升使用效率。作業系統在同一時間會存放許多工作在記憶體中，之後再從中挑選工作出來執行，但是此工作可能需要等待其他硬體I/O，故會造成CPU的閒置，此時CPU會再去記憶體中執行其他的工作，當先前的工作需要CPU時，才會取得CPU的控制權。

系統內允許多個工作行程以並行的方式執行，藉此提升CPU的使用效率。

3. 分時系統（Time-Sharing system）

批次系統和多重系統雖然可以提升效率，但是無法和使用者做雙向的互動。分時系統讓CPU在許多工作之間快速的切換並執行工作，由於切換速度很快，使用者還是可以和程式做互動，不會有卡鈍的現象。此系統強調和使用者之互動，故

4. 多處理器系統（Multiprocessing）

為現今大多數系統之類型。此種系統擁有一個以上的處理器，彼此緊密溝通，並共用電腦匯流排（Bus）、時脈（Clock）、記憶體（Memory）和I/O設備。其主要優點為：

(1) 提高產出：處理器數目的增加有助於在較短的時間內完成較多的工作。

但是處理器的數目為n時，不一定代表速度提升n倍，主要的原因為當多個不同處理器共同處理一件工作時，必定會產生額外的負荷來確保工作的進行，再加上資源競爭的影響，造成無法倍增的情況。

(2) 規模經濟：若許多程式在處理資料時可以共用資源、硬碟資料...等，就可以節省經費與存取資源。

5. 分散式系統

透過網路將許多電腦資源放置在不同地方，使得多部處理器可以互相溝通與配合，常見的架構有對等式（peer to peer）與主從式（client to server）。其特點為：

(1) 提升處理速度

(2) 達到共享資源

(3) 提升可靠性，同時也具有容錯的能力

(4) 處理器間具有溝通的特性

6. 下列何者資訊安全之敘述為錯誤？
(A) 暴力破解法：利用工具來列舉所有可能的字元組合，用以破解密碼。
(B) 字典攻擊法：利用字典的字做為基礎，用來測試使用者的密碼是否相符於字典中的字。
(C) 社交工程：攻擊者利用手段欺騙受害者，使得攻擊者取得授權來存取某特定的資源。
(D) 彩虹表：彩虹表內含所有可能的字串組合以及其相對應的 hash 值，可以用來快速查詢密碼。
(E) 以上皆為正確。

【講義命中】計算機概論第二回 P.31，相似度 70%

(五) 加密技術破解種類

1. 只知密文破解（Ciphertext Only Attack）
攻擊者盡可能蒐集所有密文以找出金鑰或明文。

2. 已知明文破解（Known Plaintext Attack）
攻擊者透過已知的明文與密文來找出金鑰。

3. 選擇明文破解（Chosen Plaintext Attack）
攻擊者將明文發送給加密伺服器，再取得密文，並用此方法設法得到金鑰。

4. 選擇密文破解（Chosen Ciphertext Attack）
與選擇明文破解方式相反，攻擊者將密文發送給解密伺服器，再透過取得的明文與已知的密文推出金鑰。

5. 暴力破解法（Brute-Force Attack）
嘗試所有可能之組合來取得金鑰或攻擊系統。

8. 以下關於物件導向程式語言之敘述，何者錯誤？
(A) 封裝（Encapsulation）的目的是將程式碼切割成許多類別，使其之間的關連性降到最低。
(B) 繼承（Inheritance）的目的是要達到「程式碼再用」（Code Reuse）或「介面再用」。
(C)多型（Polymorphism）指的是使用同一個操作介面，以操作不同的物件實例（Instance），降低對操作介面的依賴程度，進而增加程式架構的彈性與可維護性。
(D) Overload（多載）與 Override（重載）皆為實踐多型的技術之一。
(E) 以上皆正確。

【講義命中】計算機概論第三回 P.50~P51，相似度 90%

三、物件導向三大特性

(一)封裝（Encapsulation）

目的為讓使用者不需要知道物件的內部是如何運作的只需要透過操作介面就可以使用該物件的方法。將資料封裝成一個類別後，其內的成員函式與變數可以被設定成私有的（private）或公開的（public），確保類別內的變數和函數，能被我們正確的存取和使用。

- 資訊隱藏（Information Hiding）：將物件實作的方法隱藏，只提供介面給使用者操作該物件。
- 私有的：不對外公開，只能在物件內存取，若沒有指定存取控制時，預設是私有的。
- 公開的：在程式中的任何地方都可以存取。

(二)繼承（Inheritance）

定義一個類別時，可以以入其他類別的定義、方法與屬性，此特性就叫做繼承，是一種「is-a」的概念。有繼承的關係後，父類別 (Super class) 中的資料 (Data) 或方法 (Method) 在子類 (Subclass)就可以繼承使用，子類別的子類別也可以繼承使用，最後即能達到資料重覆使用的目的。

多了一個protected的存取控制，只有在衍生類別（子類別），才可以被取用。

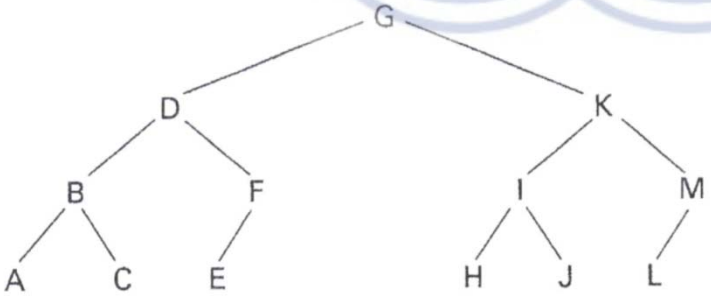
```
Class Animal
{
public:
    int age;

protected:
    bool isConservation;
```

(三)多型（Polymorphism）

多型操作指的是使用同一個操作介面，以操作不同的物件實例，多型操作在物件導向上是為了降低對操作介面的依賴程度，進而增加程式架構的彈性與可維護性。

如下例，只要Dog和Cat有實作共通的Animal介面，則兩個物件都可以使用eat這個方法。

9. 一資料結構如下，請問下列選項何者錯誤？



(A) 樹 G 為 Binary Tree，且節點 A 與節點 C 互為 sibling。
(B) 子樹 D 的 BFS（Breadth First Search）之順序為 D、B、F、A、C、E。
(C) 子樹 K 的 Inoder Traversal 順序為 H、I、J、K、L、M。
(D) 若以一連續陣列儲存整棵樹 G，則需要配置 13 單位之空間。
(E) 以上皆為正確。

【講義命中】計算機概論第四回 P.47~P.49，觀念命中，唯獨數值不一樣

二、二元樹（Binary Tree）

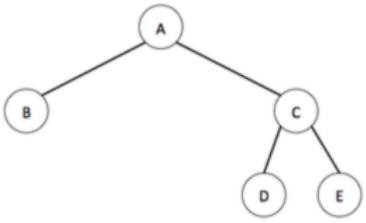
(一)基本定義

可以是空的樹或是有一個樹根極兩顆子樹，分別為左子樹與右子樹。相對於一般的樹具有以下兩個差異：

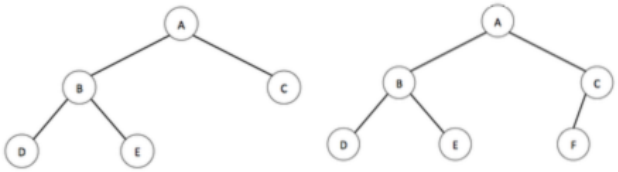
1. 二元樹可以為空，樹不行。
2. 二元樹的子樹是有順序的（Ordered）。

(二)特色

1. 若二元樹的階層為i，則其節點數最多為2^i-1。
2. 完滿二元樹（Full Binary Tree）：深度為i且具有2^i-1個節點的二元樹。



3. 完全二元樹（Complete Binary Tree）：深度為i的二元樹，若第1到第i-1層節點全滿，第i層只缺最右側若干節點（也可不缺）。



4. 嚴格二元樹（Strictly Binary Tree）：二元樹除了樹葉之外，每個節點皆同時具有非空左子樹與右子樹。
5. 歪斜樹（Skewed Binary Tree）：除了樹葉之外，每個節點皆只有右子樹，稱為右斜樹（Right-Skewed Binary Tree）。若只有左子樹，則稱為左斜樹（Left-Skewed Binary Tree）。
6. 若一二元樹有n個節點，B代表樹的總分支數，n_i代表分支度為i的節點數，則會滿足以下關係式：

   $B = n - 1$

   $n = n_0 + n_1 + n_2$

   $n_0 = n_2 + 1$

(四)二元樹的走訪

1. 中序走訪（Inorder Traversal）：先走訪左子樹，再走訪樹根，最後走訪右子樹。

```
void inorder(TreePtr tree) {
      if (tree != NULL) {
            inorder (t->lChild);
            printf("%d", t->data);
            inorder(t->rChild);
      }
}
```

2. 前序走訪（Preorder Traversal）：先走訪樹根，再走訪左子樹，之後走訪右子樹。

```
void preorder(TreePtr tree) {
      if (tree != NULL) {
            printf("%d", t->data);
            preorder (t->lChild);
            preorder(t->rChild);
      }
}
```

17. 以下關於作業系統處理死結（deadlock）之敘述何者正確？

(A) 死結問題一定是無解的。

(B) 作業系統無法預估死結問題存在。

(C) 單人單工的系統不會發現死結。

(D) 某資源若被多個程序共享，則一定會發生死結。

(E) 以上皆錯誤。

【講義命中】計算機概論第五回 P.27~P.28，相似度 80%

第三節　死結（Deadlock）

一、基本定義

指系統中存在一個以上的行程陷入互相等待對方所持有的資源，造成所有行程無法繼續往下執行的情況，使得CPU使用率大幅降低。

其發生必須符合以下四個條件：

1. 互斥（Mutual Exclusion）：
   某些特定的資源在同一個時間點最多只能被一個行程所使用。

2. 持有並等待（Hold and Wait）：
   某行程持有部分資源，並等待其他行程正在持有的資源。

3. 不可搶奪（No preemptive）：
   行程間不可以任意搶奪其他行程所持有的資源。

4. 循環等待（Circulatr Waiting）：
   系統存在一組行程：P_0, P_1, ……, P_n，其中P_0等待P_1所持有的資源，P_n等待P_0所持有的資源，形成循環等待。

死結	飢餓現象
1. 由於一組行程形成Circular waiting，導致所有行程都無法繼續執行下去。 2. CPU的使用率與Throughput會大幅下降。 3. 在Non-preemptive的情況中最容易發生。 4. 三大解決方法（下述）	1. 由於少數行程因長期取不到資源所造成，但是其他行程仍可以正常運行。 2. CPU的使用率與Throughput不一定會大幅下降。 3. 異發生在不公平的排班技術與Preemptive的環境中。 4. 解決方法：Aging技術。