



Chapter [4]

函數function

我們可以把複雜的敘述區塊包含在函數裡，需要時再呼叫函式，就可以把這複雜的區塊載入。這樣可以讓我們加快軟體的開發，並使撰寫程式簡單多了。

在 JAVA 中已經定義好的類別函數稱為類別靜態函數，這些類別靜態 static 函數都是 JAVA 事先已經寫好只要我們呼叫他就可載入我們程式執行。而我們也可以自己定義函數，這就稱為使用者自訂函數。

我們也可以自己定義類別靜態函數，這就稱為類別靜態函數。

語法：

```
修飾子 static 回傳資料型態 函數名子(參數1,參數2,參數3,,,,)
{
    敘述區塊(statement);
}
```

當我們要使用函數時，只要在該類別呼叫函數就可以了。或者接類別名稱.靜態函數();。

範例 Sqrt.Java

第三行使用sqrt()函數回傳將所輸入的數值開根號。Math是數學物件，sqrt()函數是Math類別的靜態方法。

第四行的println()函數會將我們所輸入的參數輸出。

```
1 public class sqrt{
2     public static void main(String[] args){
3         double i=Math.sqrt(4);
4         System.out.println(i);
5     }
6 }
7 }
```

4開根號後得2.0。



```
C:\Program Files\Xinox Software\JCreator\J3\LE\GE2001.exe
2.0
```

範例 Gcd.Java

Function gcd() 函數為我們自訂的函數，這是求數學最大公因數，我們在使用時，只要呼叫他就可以了。例如我們使用gcd(21,15)，來呼叫gcd() 函數，並且將參數21，15帶入，而在運算後他就會回傳return答案。

```

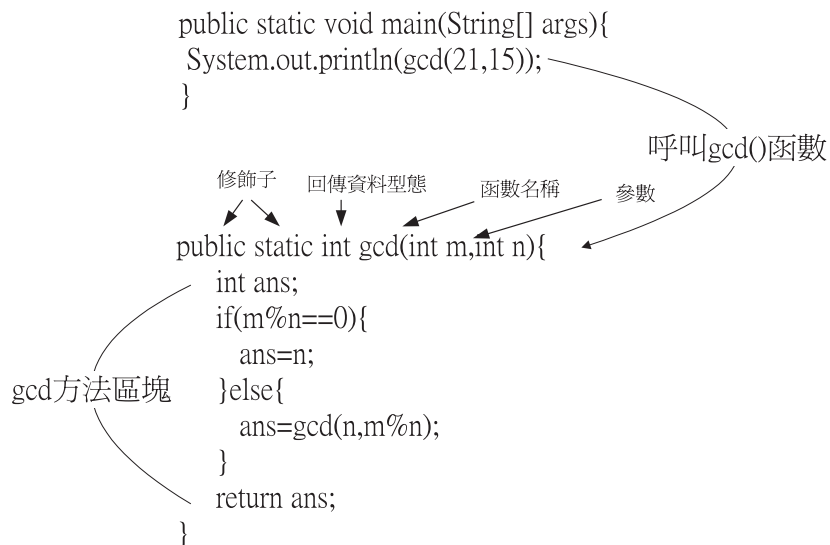
1 public class Gcd{
2     public static void main(String[] args){
3         System.out.println(gcd(21,15));
4     }
5     public static int gcd(int m,int n){
6         int ans;
7         if(m%n==0){
8             ans=n;
9         }else{
10            ans=gcd(n,m%n);
11        }
12        return ans;
13    }
14 }

```

顯示回傳值return value，21和15的最大公因數是3




我們使用gcd(21,15)來呼叫gcd()函數。



這是Gcd類別。當載入該類別Gcd到記憶體時，程式會先執行main()函數，當主程式在呼叫gcd()類別函數時，執行的程式就會跳到類別函數的記憶體敘述區塊，然後執行敘述區塊，當執行完後再跳回程式主體繼續往下執行。宣告成static的函數為類別函數，也就是再該類別被載入到記憶體時，類別static靜態函數也會被載入，而且只有一份copy拷背。

主記憶體



```
public class Gcd{
    public static void main(String[] args){
        System.out.println(gcd(21,15));
    }
    public static int gcd(int m,int n){
        int ans;
        if(m%n==0){
            ans=n;
        }else{
            ans=gcd(n,m%n);
        }
        return ans;
    }
}
```

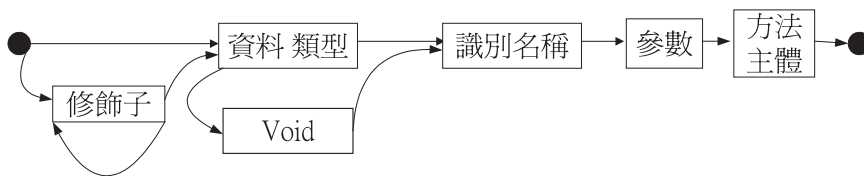
4-1 使用者自訂類別函數

我們也可以自己定義在該類別的靜態函數，這就稱為使用者自訂類別靜態函數。所有類別函數都要被宣告成 `static` 靜態。

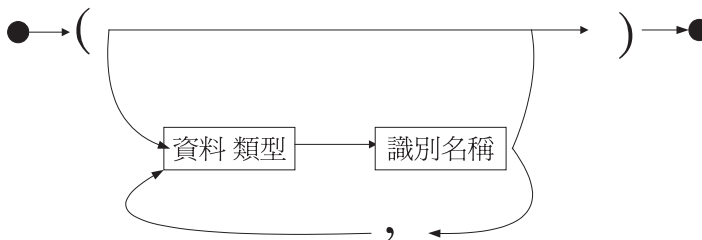
語法：

```
修飾子 static 回傳資料型態 函數名子(參數 1,參數 2,參數 3,,,,)
{
    敘述區塊(statement);
    return 回傳資料;
}
```

這是方法宣告的語法，第一個是存取控制修飾子，像 `public`、`private`、`protected` 等，後面再接資料類型 (`void` 沒有回傳值)，再接函數的識別名稱，再接參數，最後再接方法區塊。



這是參數的語法。使用括號 () 包起來。括號內的為資料類型和資料的名稱，然後加逗號，再加另外一個參數。



當我們要使用函數時，只要在該類別呼叫類別函數就可以了。

範例 Square_area.java

我們在一開始使用了自己定義的類別函數來求最大公因數，如果我們要計算正方形面積也可以寫一個面積函數，只要輸入邊長就可以求得面積，再利用return回傳面積。

第六行到第十行我們定義了Square_area類別函數area()正方形面積函數。

第九行使用return來回傳正方形面積。

第三行呼叫自己定義的類別函數area()，並且把邊長為5的參數帶入。

```
1 public class Square_area{
2     public static void main(String[] args){
3         double j=area(5);
4         System.out.println(" 正方形的面積:"+j);
5     }
6     public static double area(double a){
7         double result;
8         result=a*a;
9         return result;
10    }
11 }
```

顯示25。



正方形的面積:25.0

4-2 函數的參數

在 Square_area.java 中，我們將邊長當 area() 函數的參數傳入 area() 函數，經過運算後傳回正方形面積。這就是 JAVA 預設的傳值呼叫 (passing by value)。在 JAVA 中使用者定義函數是傳值呼叫 (call by value)。當我們呼叫函數並傳給它變數時，JAVA 會拷貝該變數的值，並且傳送給被呼叫的函數。所以無論被呼叫的函數怎麼作，都無法改變該變數真正的值。

4-2-1 傳值呼叫

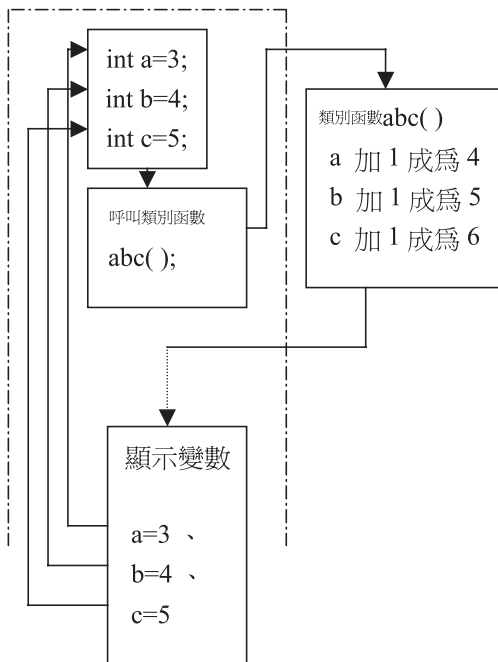
這就是 Java 預設的傳值呼叫 (passing by value)。在 JAVA 中使用者定義函數是傳值呼叫 (call by value)。當我們呼叫函數並傳給它變數時，JAVA 會拷貝該變數的值，並且傳送給被呼叫的函數。所以無論被呼叫的函數怎麼作，都無法改變該變數真正的值。

範例 Byvalue.java

第六行我們呼叫abc(a,b,c)函數，並且將a=3、b=4和c=5的值帶入。

第十一行到第十五行我們定義了abc()函數。

```
1 public class Byvalue{
2     public static void main(String[] args){
3         double a=3;
4         double b=4;
5         double c=5;
6         abc(a,b,c);
7         System.out.println("a:"+a);
8         System.out.println("b:"+b);
9         System.out.println("c:"+c);
10    }
11    public static void abc(double a,double b,double c){
12        a++;
13        b++;
14        c++;
15    }
16 }
```



虛線裡面所表示的為主程式，變數a、變數b、變數c不會受到影響，函數改變的只是複製堆疊的值，而不會影響主程式變數的值，因此顯示沒有被改變的值。

```
C:\Program Files\Xinox Software\Ucreator\3LE\GE2001.exe
a:3.0
b:4.0
c:5.0
```

範例 callbyvalue_subtract.java

這是個傳值呼叫的範例。

函數subtract_each(a,b)是定義在第八行到第十五行之間，在第十一行的變數不是在主程式上的變數，而是複製的堆疊變數。函數中每個參數都是複製變數的堆疊stack，不會影響到原本變數的值。

第五行和第六行傳值呼叫，其所傳入的參數都為a=5和b=3，因此其結果都一樣，不會修改到變數a和變數b的數值。

```
1 public class callbyvalue_subtract{
2     public static void main(String[] args){
3         double a=5;
4         double b=3;
5         System.out.println("兩個參數相減:"+subtract_each(a,b));
6         System.out.println("兩個參數相減:"+subtract_each(a,b));
7     }
8     public static double subtract_each(double a,double b){
9         double return_value=0;
10        while(a>b){
11            a=a-1;
12            return_value=return_value+1;
13        }
14        return(return_value);
15    }
16 }
```

這是執行的結果。



兩個參數相減:2.0
兩個參數相減:2.0

4-3 Overloading 過載

相同名稱的函數，但是他們所帶入的參數不相同，Java 編譯器將以函數的簽名來決定使用哪個函數，這就是方法的過載 method overloading。Java 使用簽名來辨別函數。簽名 signature 包含了函數的名稱和參數。在 Java 的類別中，可能有多個相同名稱的函數，但是它們所帶的參數個數或型態不同，這時簽名就不同，這就是同名異式。假如我們呼叫範例 Overloading_max.java 的 max 函數並且代入的參數是浮點數 double，則這個參數為浮點數的 max(double a,double b) 函數將自動被啟動。

```
public static double max(double a,double b){  
    if(a>b)  
        return a;  
    else  
        return b;  
}
```

假如我們呼叫範例 Overloading_max.java 的 max 函數並且代入的參數是整數 int，則這個參數為整數的 max(int a,int b) 函數將自動被啟動。雖然 max(double a,double b) 函數和 max(int a,int b) 函數有相同的函數名稱，但它們的簽名 (名稱和參數) 不同，所以所執行的敘述也不同。

```
public static int max(int a,int b){  
    if(a>b)  
        return a;  
    else  
        return b;  
}
```

範例 Overloading_max.java

第七行到第十二行是max()函數，帶入的是double浮點數的兩個參數。

第十三行到第十八行是max()函數，帶入的是int整數的兩個參數。

第十九行到第二十一行是max()函數，帶入的是double浮點數的三個參數。

第三行呼叫max(5,1)函數，並且將5和1的兩個整數參數帶入，Java編譯器會決定由第十三行到第十八行的max()函數執行。

第四行呼叫max(5.2,1.3)函數，並且將5.2和1.3的兩個浮點數參數帶入，Java編譯器會決定由第七行到第十二行的max()函數執行。

第五行呼叫max(0.5,1,2.5)函數，並且將0.5、1和2.5的三個浮點數參數帶入，Java編譯器會決定由第十九行到第二十一行的max()函數執行。

```
1 public class Overloading_max{
2     public static void main(String[] args){
3         System.out.println("最大值:"+max(5,1));
4         System.out.println("最大值:"+max(5.2,1.3));
5         System.out.println("最大值:"+max(0.5,1,2.5));
6     }
7     public static double max(double a,double b){
8         if(a>b)
9             return a;
10        else
11            return b;
12    }
13    public static int max(int a,int b){
14        if(a>b)
15            return a;
16        else
17            return b;
18    }
19    public static double max(double a,double b,double c){
20        return(max(max(a,b),c));
21    }
22 }
```

這是執行的情況。

```
C:\Program Files\Xinox Software\Creator\3LE\GE2001.exe
最大值:5
最大值:5.2
最大值:2.5
```

4-4 回傳return

我們使用 `return` 將函數的值回傳。

範例 Ca.Java

第一行我們輸入 `java.util.Scanner` 套件，這樣我們就可以使用 `Scanner` 物件掃描我們所輸入的參數。

第五行我們新增 `Scanner` 類別的物件 `scan`。

第八行使用 `scan.nextDouble(System.in)` 函數來得到使用者從 `System.in` 所輸入的浮點數。

第十九行我們使用 `return pay` 將函數 `finace()` 的值回傳，它回傳的為浮點數 `double` 的值。

第十五行到第二十行我們定義 `finance()` 函數，並且設定它的回傳資料型態為 `double`。

第十三行顯示所傳回來的值，我們直接呼叫函數 `finance()` 並且將參數 `pay`、`n` 和 `yr_rate` 帶入，這樣就可以得到本利合，並將其值回傳了。

```
1 import java.util.Scanner;
2 public class Ca{
3     public static void main(String[] args){
4         double pay,n,yr_rate;
5         Scanner scan=new Scanner(System.in);
6         System.out.println("定期存款,複利法計算本利合");
7         System.out.println("請輸入本金:");
8         pay=scan.nextDouble();
9         System.out.println("請輸入年利率:");
10        yr_rate=scan.nextDouble();
11        System.out.println("請輸入幾年後:");
12        n=scan.nextDouble();
13        System.out.println("本利合:"+finance(pay,n, yr_rate));
14    }
15    public static double finance(double pay,double n,double yr_rate){
16        for(int i=1;i<=n;i++){
17            pay=pay*(1+yr_rate/100);
18        }
19        return pay;
20    }
21 }
```

本金1000，10%利率，10年複利的回傳值是2593.74元。

C:\Program Files\Xinox Software\JCreator\Y3LE\GE2001.exe

定期存款，複利法計算本利合

請輸入本金：

1000.5

請輸入年利率：

10

請輸入幾年後：

10

本利合：2595.039331330052

4-5 區域變數的生存空間

只有作用在函數內或大括號“{”和“}”的稱為區域變數。區域變數存在的範圍是從區域變數宣告的地方開始，而到包含該變數區塊結束的地方。區域變數一定要先宣告它才能使用它。

函數的參數為區域變數。函數的參數的生存空間只在函數的範圍內，或大括號“{”和“}”內的範圍。

當在 for 迴圈的初始化中宣告變數時，它的生存空間就在整個的迴圈中。當在 for 迴圈內部的區塊中宣告變數時，它的生存空間就只在它宣告的地方到區塊結束的地方為止。在大括號 {} 裏面的變數都是區域變數。

```
public static void method(){
    .
    .
    for(int i=1;i<5;i++){
        .
        .
        int j;
        .
    }
}
```

變數能在非巢狀的區塊被宣告好幾次，但最好只在巢狀的區塊宣告一次。

```
public static void method1(){
    int a=5;
    int b=5;
    .
    for(int i=1;i<5;i++){
        .
    }
    for(int i=1;i<5;i++){
        .
    }
}
```

在兩個非巢狀的區塊中宣告變數int是可以的

```
public static void method2(){
    int i=1;
    .
    for(int i=1;i<5;i++){
        .
    }
}
```

在巢狀的區塊宣告變數i兩次是不好的變數宣告

4-6 遞迴函數recursive

我們在函數的開始就介紹了最大公因數的求法，他就是使用遞迴函數。遞迴函數就是自己呼叫自己函數的意義。對於某一些難以解決的問題，遞迴提供了一個自然、簡單的解決方案，因此遞迴真的是很有用的方法。

範例

Mul.java

這是遞迴的例子，我們求得8乘3得24就是執行第十一行的函數multiply(8,3)將3個8相加。

第一行我們輸入java.util.Scanner套件，這樣我們就可以使用Scanner物件掃描我們所輸入的參數。

第五行我們新增Scanner類別的物件scan。

第八行使用scan.nextDouble(System.in)函數來得到使用者從System.in所輸入的浮點數。

第十一行顯示所傳回來的值，我們直接呼叫函數multiply()函數並且將參數m和n帶入，這樣就可以得到積，並將其值回傳了。

第十三行到第二十一行我們定義multiply()函數，並且設定它的回傳資料型態為double。

第二十行我們使用return ans將函數multiply()的值回傳，它回傳的為浮點數double的值。

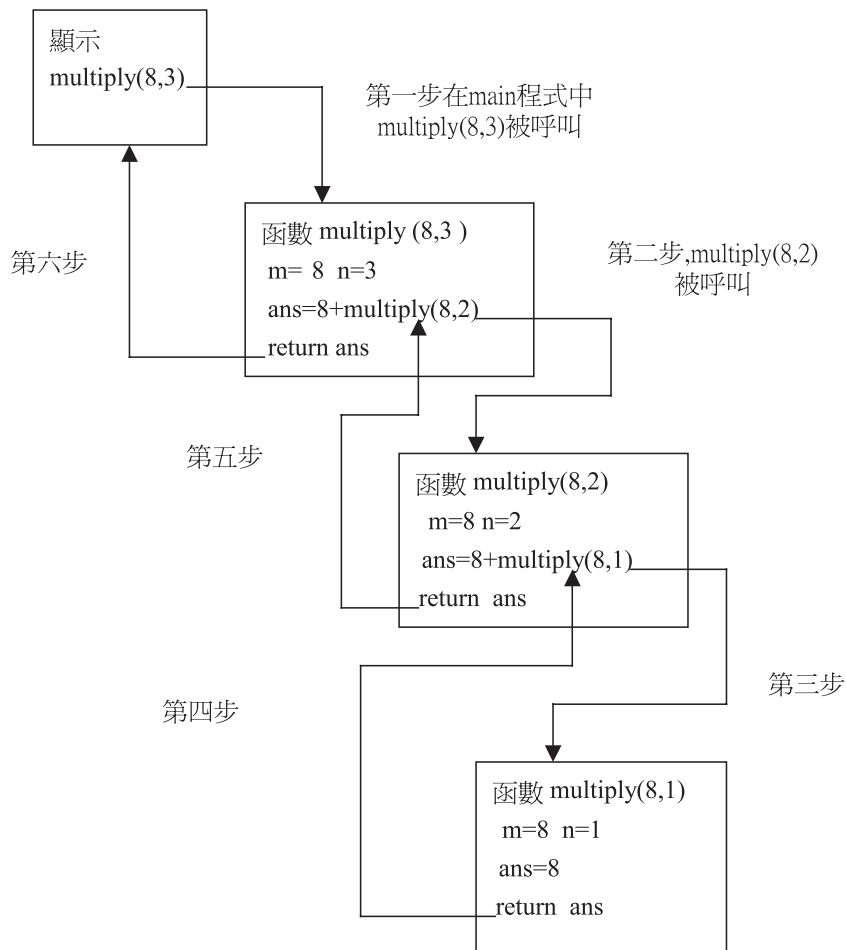
```
1 import java.util.Scanner;
2 public class Mul{
3     public static void main(String[] args){
4         double m,n;
5         Scanner scan=new Scanner(System.in);
6         System.out.println("m*n乘法運算");
7         System.out.println("請輸入m:");
8         m=scan.nextDouble();
9         System.out.println("請輸入整數n:");
10        n=scan.nextDouble();
11        System.out.println("m*n="+multiply(m,n));
12    }
13    public static double multiply(double m,double n){
14        double ans;
15        if(n==1){
16            ans=m;
17        }else{
18            ans=m+multiply(m,n-1);
19        }
20        return ans;
21    }
22 }
```



我們可以用遞迴解決許多問題。這是遞迴的例子，我們求得8乘3得24就是執行第十一行的函數multiply(8,3)將3個8相加。

```
C:\Program Files\Xinox Software\CreatorV3LE\GE2001.exe
m*n 乘法運算
請輸入m:
8
請輸入整數n:
3
m*n=24.0
```

這一題顯示了函數自己呼叫自己(遞迴函數)而且也說明了函數傳回值return value得情況。



範例 Factorial.java

這是factorial()函數，可以求n!的數值。遞迴函數factorial()在第十六行呼叫自己，並且將n - 1的值帶入遞迴函數。這樣的遞迴數學式就是 $N * (N - 1) * ((N - 1) - 1) * \dots * 1$ 。

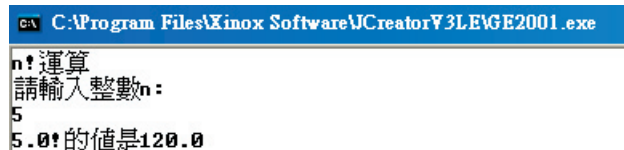
遞迴函數有兩個條件。一個是最終遞迴的值，然後開始回傳；一個就是遞迴函數的運算式。這兩個條件就可以組成遞迴函數。Factorial()最終的值是ans=1；遞迴函數的運算式是ans=n*factorial(n-1)。

```

1 import java.util.Scanner;
2 public class Factorial{
3     public static void main(String[] args){
4         double m,n;
5         Scanner scan=new Scanner(System.in);
6         System.out.println("n!運算");
7         System.out.println("請輸入整數n:");
8         n=scan.nextDouble();
9         System.out.println(n+"!的值是"+factorial(n));
10    }
11    public static double factorial(double n){
12        double ans;
13        if(n==0){
14            ans=1;
15        }else{
16            ans=n*factorial(n-1);
17        }
18        return ans;
19    }
20 }

```

這是執行的情況。



```

C:\Program Files\Xinox Software\JCreator\J3LE\GE2001.exe
n!運算
請輸入整數n:
5
5.0!的值是120.0

```

範例 Fibonacci.java

這是費氏系數的問題。費氏級數是由0和1開始，每一個費氏級數的數，是由前兩個費氏級數的合所組成。

費氏級數 0 1 1 2 3 5 8 13 21 34 55

索引 0 1 2 3 4 5 6 7 8 9 10



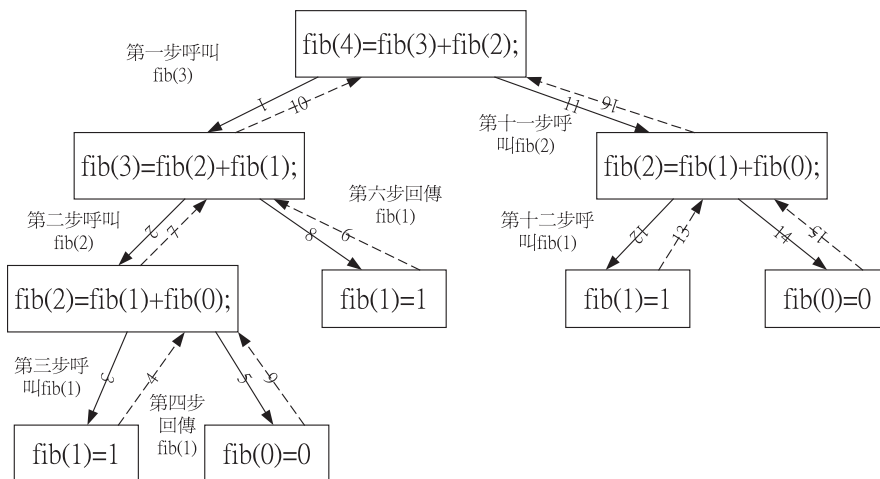
我們可以將費氏級數定義成。

```
fib(0)=0 ;
fib(1)=1 ;
fib(index)=fib(index -1)+fib(index-2) ;
```

遞迴函數有兩個條件。一個是最終遞迴的值，然後開始回傳；一個就是遞迴函數的運算式。這兩個條件就可以組成遞迴函數。fib()最終的值是fib(0)=0和fib(1)=1；遞迴函數的運算式是fib(index)=fib(index -1)+fib(index-2)；。

```
if(n==0){
    ans=0;
}else if(n==1){
    ans=1;
}else{
    ans=fib(n-1)+fib(n-2);
}
```

當我們要求解費氏級數4時的圖解。



第一行我們輸入java.util.Scanner套件，這樣我們就可以使用Scanner物件掃描我們所輸入的參數。

第五行我們新增Scanner類別的物件scan。

第八行使用scan.nextDouble(System.in)函數來得到使用者從System.in所輸入的浮點數。

第九行顯示所傳回來的值，我們直接呼叫費氏fib()函數並且將參數n帶入，這樣就可以得到費氏級數，並將其值回傳了。

第十一行到第二十一行我們定義費氏fib()函數，並且設定它的回傳資料型態為double。

第二十行我們使用return ans將費氏函數fib()的值回傳，它回傳的為浮點數double的值。

```
1 import java.util.Scanner;
2 public class Fibonacci{
3     public static void main(String[] args){
4         double m,n;
5         Scanner scan=new Scanner(System.in);
6         System.out.println("費氏系數運算");
7         System.out.println("請輸入整數n:");
8         n=scan.nextDouble();
9         System.out.println(n+"的費氏系數是"+fib(n));
10    }
11    public static double fib(double n){
12        double ans;
13        if(n==0){
14            ans=0;
15        }else if(n==1){
16            ans=1;
17        }else{
18            ans=fib(n-1)+fib(n-2);
19        }
20        return ans;
21    }
22 }
```

這是費氏級數4的解答4。

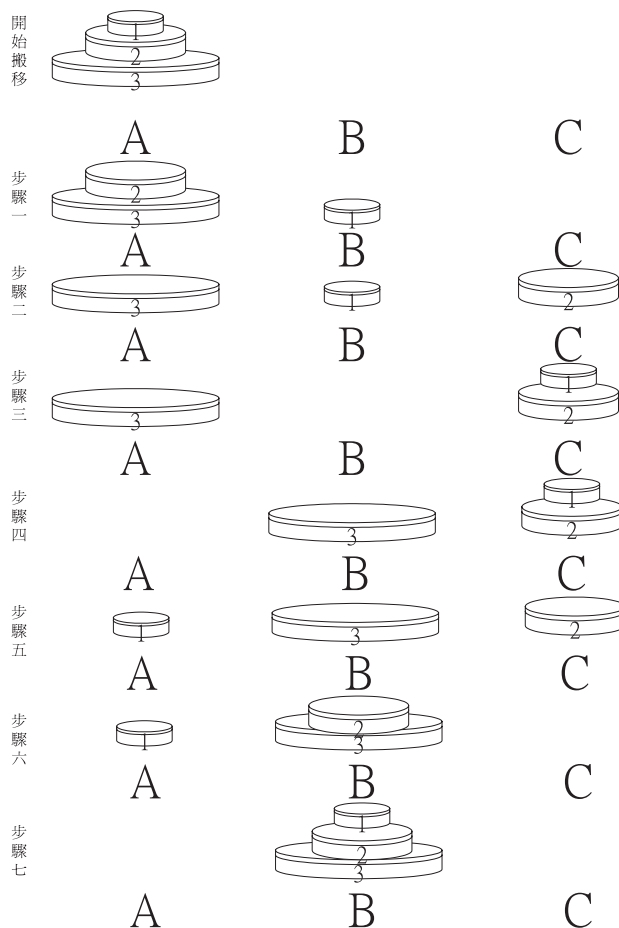
```
C:\Program Files\Xinox Software\JCreator\3LE\GE2001.exe
費氏系數運算
請輸入整數n:
4
4.0的費氏系數是3.0
```

4-6-1 河內塔的遞迴問題

我們要移動數個盤子從 A 點移動到 B 點，有下列的規則。

1. 有n個標籤的盤子分別為1、2、3、4、5...到n，然後有三個塔分別為A、B和C。
2. 在任何時刻，大盤子都要比小盤子位於低處。
3. 所有的盤子一開始都是在A塔。
4. 一次只能搬一個盤子，而且小盤子一定要在大盤子上面。

我們要將盤子從 A 塔搬到 B 塔，可以介由 C 塔的幫助。我們首先將 1 號盤子從 A 移到；再將 2 號盤子從 A 移到 C；再將 1 號盤子從 B 移到 C；再將 3 號盤子從 A 移到 B；再將 1 號盤子從 C 移到 A；將 2 號盤子從 C 移到 B；最後將 1 號盤子從 A 移到 B，這樣就結束了。



如果我們有大量的盤子要移動，我們需要演算法來協助我們運算。

遞迴函數河內塔的最終條件就是當盤子為一個時 $n=1$ ，我們只要將盤子從 A(from 塔) 移到 B(to 塔) 就可以了。當 $n>1$ 時，我們可以將問題分解成三個子問題。這樣遞迴的解決了河內塔的所有問題。

1. 移動這首先的n-1個盤子從A(from塔)到C(aux塔)，然後用B(to塔)的協助。

(步驟一到步驟三)

2. 移動盤子n，從A(from塔)到B(to塔)。

(步驟四)

3. 移動這n-1個盤子，從C(aux塔)到B(to塔)，然後用A(from塔)的協助。

(步驟五到步驟七)

這是程式碼。

```
if(n==1){
    System.out.println(" 移動盤子"+n+" 從"+from+" 塔到"+to+" 塔");
}else{
    move(n-1, from, aux, to);
    System.out.println(" 移動盤子"+n+" 從"+from+" 塔到"+to+" 塔");
    move(n-1, aux, to, from);
}
```

移動 move() 函數，這就是移動 n 個盤子，從這 from 塔到 to 塔，然後使用 aux 塔的協助。

move(int n,char from,char to,char aux)

範例

Hanoi.java

第十行開始移動n個盤子，從A塔到B塔，使用C塔的協助。

第十三行和第十四行，最終條件當n==1時，移動盤子1從from塔到to塔。

第十六行移動這首先的n-1個盤子從from塔到aux塔，然後用to塔的協助。

第十七行移動盤子n，從from塔到to塔。

第十八行移動這n-1個盤子，從aux塔到to塔，然後用from塔的協助。

```

1 import java.util.Scanner;
2 public class Hanoi{
3     public static void main(String[] args){
4         int n;
5         Scanner scan=new Scanner(System.in);
6         System.out.println("河內塔移動盤子問題");
7         System.out.println("請輸入盤子個數(整數n):");
8         n=scan.nextInt();
9         System.out.println("盤子在河內塔移動過程:");
10        move(n,'A','B','C');
11    }
12    public static void move(int n,char from,char to,char aux){
13        if(n==1){
14            System.out.println("移動盤子"+n+"從"+from+"塔到"+to+"塔");
15        }else{
16            move(n-1, from, aux, to);
17            System.out.println("移動盤子"+n+"從"+from+"塔到"+to+"塔");
18            move(n-1, aux, to, from);
19        }
20    }
21 }

```

這是執行的情況。

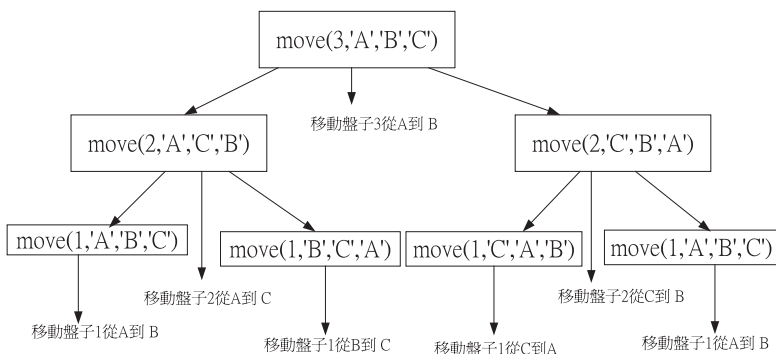
C:\Program Files\Xinox Software\JCreator\3LE\GE2001.exe

```

河內塔移動盤子問題
請輸入盤子個數<整數n>:
3
盤子在河內塔移動過程:
移動盤子1從A塔到B塔
移動盤子2從A塔到C塔
移動盤子1從B塔到C塔
移動盤子3從A塔到B塔
移動盤子1從C塔到A塔
移動盤子2從C塔到B塔
移動盤子1從A塔到B塔

```

這是三個盤子從河內塔移動的情況。這是中序的樹狀執行(資料結構)。中序尋訪在樹狀結構中向下，向左移動，一直到達空節點為止。然後拜訪此空節點的父節點，並且由它的右子節點繼續尋訪，如果不能向右移動，則由上一層中的最後一個未被拜訪的節點繼續尋訪。



習題

1. 請簡述類別函數？

【答案】

2. 請簡述傳值呼叫？

【答案】

Java預設是傳值呼叫(passing by value)。在JAVA中使用者定義函數是傳值呼叫(call by value)。當我們呼叫函數並傳給它變數時，JAVA會拷貝該變數的值，並且傳送給被呼叫的函數。所以無論被呼叫的函數怎麼作，都無法改變該變數真正的值。

3. 請用一個簡單範例來表示類別函式呼叫？

【答案】

4. 請解釋覆載Overloading？

【答案】

相同名稱的函數，但是他們所帶入的參數不相同，Java編譯器將以函數的簽名來決定使用哪個函數，這就是方法的過載method overloading。Java使用簽名來辨別函數。簽名signature包含了函數的名稱和參數。在Java的類別中，可能有多個相同名稱的函數，但是它們所帶的參數個數或型態不同，這時簽名就不同，這就是同名異式。假如我們呼叫範例Overloading_max.java的max函數並且代入的參數是浮點數double，則這個參數為浮點數的max(double a,double b)函數將自動被啟動。



5. 請簡述區域變數和它的生存空間？

【答案】

只有作用在函數內或大括號“{ “和” }”內的稱為區域變數。區域變數存在的範圍是從區域變數宣告的地方開始，而到包含該變數區塊結束的地方。區域變數一定要先宣告它才能使用它。函數的參數為區域變數。函數的參數的生存空間只在函數的範圍內或大括號“{ “和” }”內的範圍。

當在for迴圈的初始化中宣告變數時，它的生存空間就在整個的迴圈中。當在for迴圈內部的區塊中宣告變數時，它的生存空間就只在它宣告的地方到區塊結束的地方為止。在大括號{}裏面的變數都是區域變數。

6. 請簡述遞迴函數？

【答案】

我們在函數的開始就介紹了最大公因數的求法，他就是使用遞迴函數。遞迴函數就是自己呼叫自己函數的意義。對於某一些難以解決的問題，遞迴提供了一個自然、簡單的解決方案，因此遞迴真的是很有用的方法。

7. 請簡述遞迴函數自己呼叫自己的過程？

【答案】

8. 請使用遞迴函數解釋factorial()函數？

【答案】

這是factorial()函數，可以求n!的數值。遞迴函數factorial()在第十六行呼叫自己，並且將n - 1的值帶入遞迴函數。這樣的遞迴數學式就是 $N*(N-1)*((N-1)-1)*\cdots*1$ 。

遞迴函數有兩個條件。一個是最終遞迴的值，然後開始回傳；一個就是遞迴函數的運算式。這兩個條件就可以組成遞迴函數。Factorial()最終的值是ans=1；遞迴函數的運算式是ans=n*factorial(n-1)。

9. 請用遞迴函數解釋費氏系數？

【答案】

10. 請簡述河內塔的遞迴問題？

【答案】

我們要移動數個盤子從A點移動到B點，有下列的規則。

1. 有n個標籤的盤子分別為1、2、3、4、5...到n，然後有三個塔分別為A、B和C。
2. 在任何時刻，大盤子都要比小盤子位於低處。
3. 所有的盤子一開始都是在A塔。
4. 一次只能搬一個盤子，而且小盤子一定要在大盤子上面。

演算法：

遞迴函數河內塔的最終條件就是當盤子為一個時 $n=1$ ，我們只要將盤子從A(from塔)移到B(to塔)就可以了。當 $n>1$ 時，我們可以將問題分解成三個子問題。這樣遞迴的解決了河內塔的所有問題。



1. 移動這首先的 $n-1$ 個盤子從A(from塔)到C(aux塔)，然後用B(to塔)的協助。
(步驟一到步驟三)
2. 移動盤子 n ，從A(from塔)到B(to塔)。
(步驟四)
3. 移動這 $n-1$ 個盤子，從C(aux塔)到B(to塔)，然後用A(from塔)的協助。
(步驟五到步驟七)

這是程式碼。

```
if(n==1){
    System.out.println(" 移動盤子"+n+" 從"+from+" 塔到"+to+" 塔");
}else{
    move(n-1, from, aux, to);
    System.out.println(" 移動盤子"+n+" 從"+from+" 塔到"+to+" 塔");
    move(n-1, aux, to, from);
}
```

移動move()函數，這就是移動 n 個盤子，從這from塔到to塔，然後使用aux塔的協助。

`move(int n,char from,char to,char aux)`