實作輔導

- 本周5/5(六)安排實作輔導
- 二時段:
  - 周六 11:00~12:30
  - 周六13:30~15:30

搶答!!

Q1：印出結果?

```
int x=10, y=15;
  int s=0;
  while (x>y) {
    x--;
    while (x%2==1 && s<3) s++;
   }
  System.out.println("x="+x+"s="+s);
```

Q2：印出結果?

```
x=10; y=15;
s=20;
for(;x>=5;x--)
  for(;y<=20;y++)
     s--;
System.out.println("x="+x+"y="+y+"s="+s);
```

Self-regulated learning

```
import java.util.Scanner;
public class nested_looptest1 {
 static Scanner input = new Scanner(System.in);
 public static void main(String[] args) {
  int x=10, y=15;
  int s=0;
  while (x>y) {
    x--;
    while (x%2==1 && s<3) s++;
   }
  System.out.println("x="+x+"s="+s);
  x=10; y=15;
  s=20;
  for(;x>=5;x--)
   for(;y<=20;y++)
      s--;
  System.out.println("x="+x+"y="+y+"s="+s);
                }//main
  //x=10s=0
  //x=4y=21s=14
}//class_loop1
```

迴圈LOOP應用 II

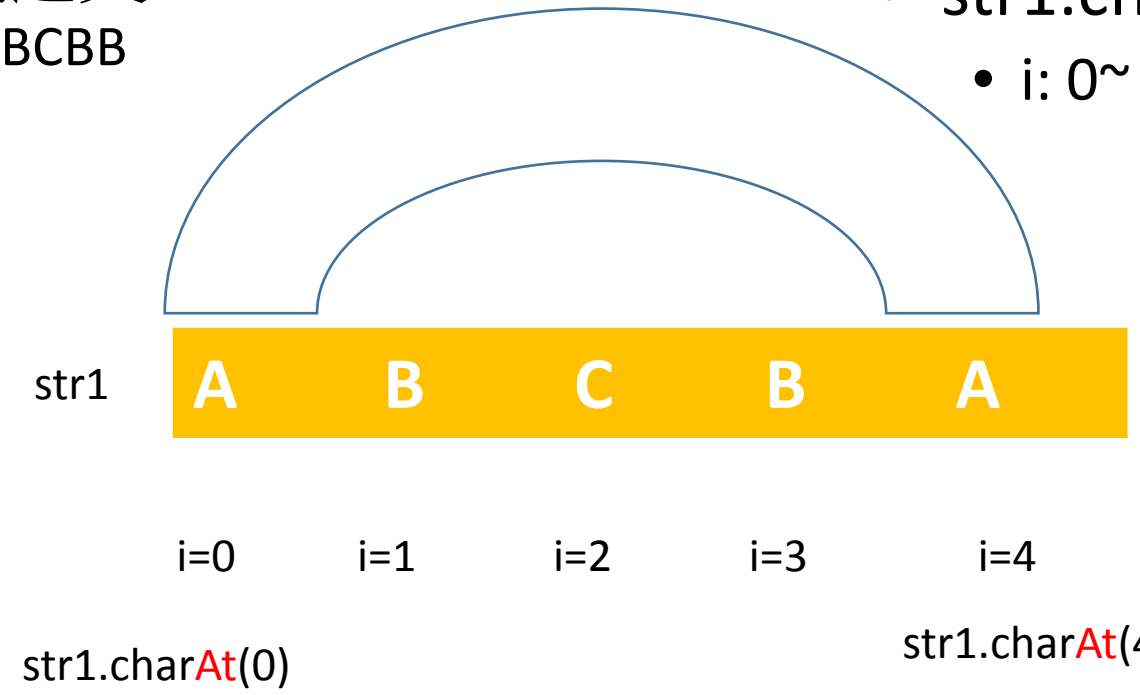
- **判斷**迴文(palindrome): 複習
- 求兩個整數的最大公因數(greatest common divisor, GCD): II
  - **輾轉相除法**
  - 輾轉相減法
- 字元金字塔 : Nested for loop

複習
palindrome迴文

- 2017/1/2 line

palindrome迴文

- 輸入字串，判斷是否為迴文?
- 迴文
  - 2017102
  - ABCBA
  - ABBA
- 不是迴文
  - ABCBB



- 輸入字串

```
Scanner input = new Scanner(System.in);
String str1 = input.nextLine();
```

- str1.length() :字串長度
- str1.charAt(i):取得字串第i個字元(character)
  - i: 0~ str1.length()-1

str1.length()==5

取得字串第i個字元(character)

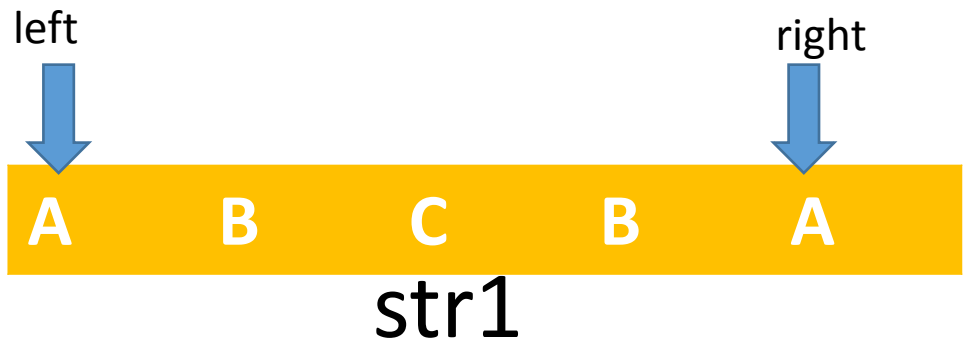
```
System.out.print("輸入字串：");
str1 = input.nextLine();
if (str1.length()==0) break;
int left=0,right=0, i=0;
boolean palindrome=false;
while (i<=str1.length()-1) {
    System.out.print(str1.charAt(i));
    i++;}

System.out.print("\n");
i=str1.length()-1;
while (i>=0) {
    System.out.print(str1.charAt(i));
    i--;}
System.out.println("\n");
```

輸入字串，判斷是否為迴文?奇數個字元為例

```
left=0;right=str1.length()-1;
boolean palindrome=true;
while (left<right) {
   if (str1.charAt(left)!=str1.charAt(right))   // !=  不等於
       {palindrome=false;
            break;}
    left++;  right--;
    }//while
if (palindrome)  dif="是迴文!“;
else                    dif="不是迴文!";
System.out.println(+dif);
```

奇數個字元

Left=0; right=4:比較str1.charAt(0)、 str1.charAt(4)
Left=1; right=3:比較str1.charAt(1)、 str1.charAt(3)
Left=2; right=2:比較str1.charAt(2)、 str1.charAt(2)
Left=3; right=1; ➔ left >right ➔ 離開loop

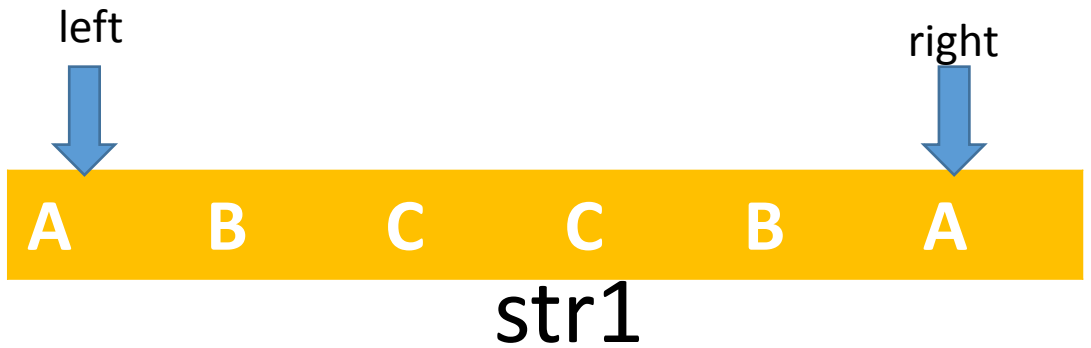


輸入字串，判斷是否為迴文?偶數個字元為例

```
left=0;right=str1.length()-1;
boolean palindrome=true;
while (left<right) {
   if (str1.charAt(left)!=str1.charAt(right))   // !=  不等於
       {palindrome=false;
           break;}
    left++;  right--;
    }//while
if (palindrome)  dif="是迴文!“;
else                    dif="不是迴文!";
System.out.println(+dif);
```

偶數個字元

Left=0; right=5: 比較str1.charAt(0)、 str1.charAt(5)
Left=1; right=4:比較str1.charAt(1)、 str1.charAt(4)
Left=2; right=3:比較str1.charAt(2)、 str1.charAt(3)
Left=3; right=2; ➔ left >right ➔ 離開loop



求兩個整數的最大公因數(greatest common divisor, GCD) :ll

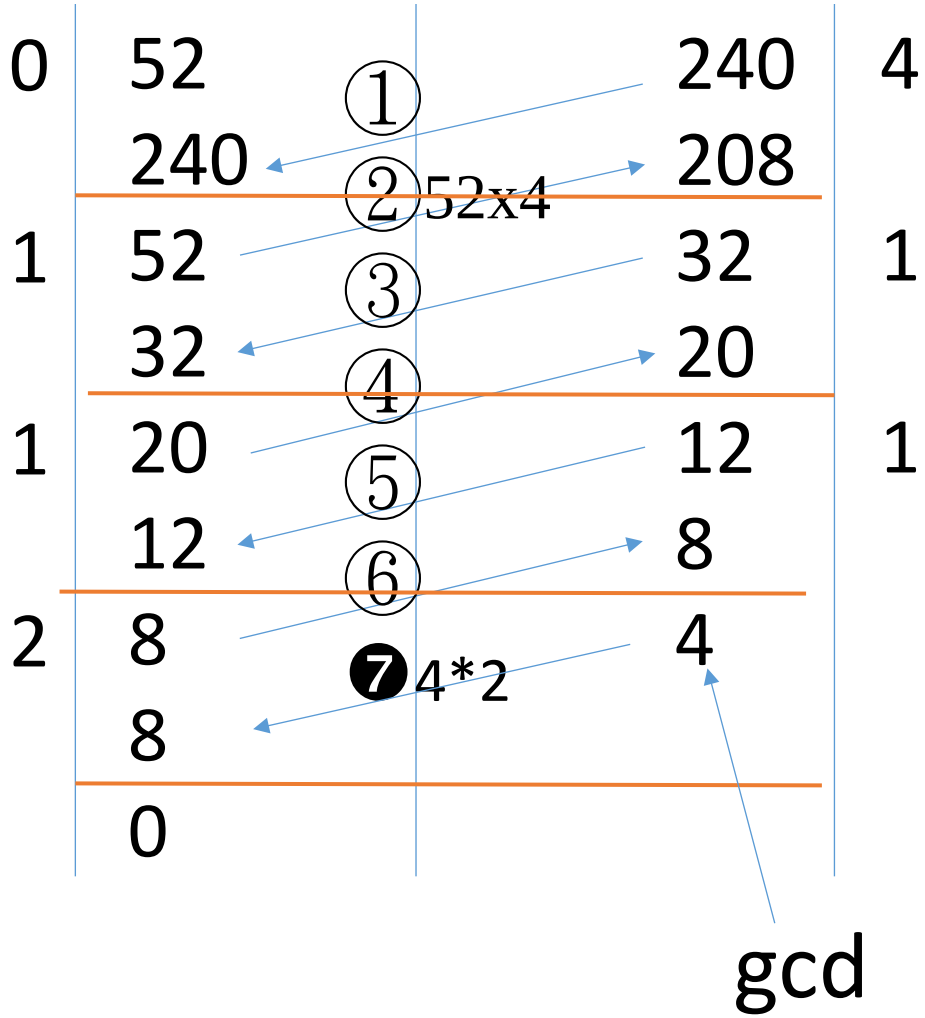
- 兩個整數的最大公因數(greatest common divisor)是能夠同時整除它們的最大的正整數
- 求兩個整數GCD的方法:
  - 從2開始找， 直到能整除兩個整數的最大正整數
    - 何時結束 (不會超過兩個整數的最小整數)
    - 200, 40的GCD
  - **輾轉相除法 (本回)**
  - 輾轉相減法 **(本回)**
- 最小公倍數(LCM): n1*n2/gcd

求兩個整數GCD的方法2: **輾轉相除法**

- **輾轉相除法**，又稱**歐幾里得算法**（Euclidean algorithm）
- 兩個整數的最大公因數(greatest common divisor)是能夠同時整除它們的最大的正整數

```
輸入整數n1 & n2,輾轉相除法:求GCD(n1, n2) and LCM(n1, n2)
輸入整數n1：11
輸入整數n2：121
輾轉相除法: GCD(11,121)=11
輾轉相除法: LCM(11,121)=121
被除數11除以除數121得餘數11
被除數變為121,除數變為11
被除數121除以除數11得餘數0
被除數變為11,除數變為0
輾轉相除法: GCD(11,121)=11
輾轉相除法: LCM(11,121)=121
```

求GCD(52,240)



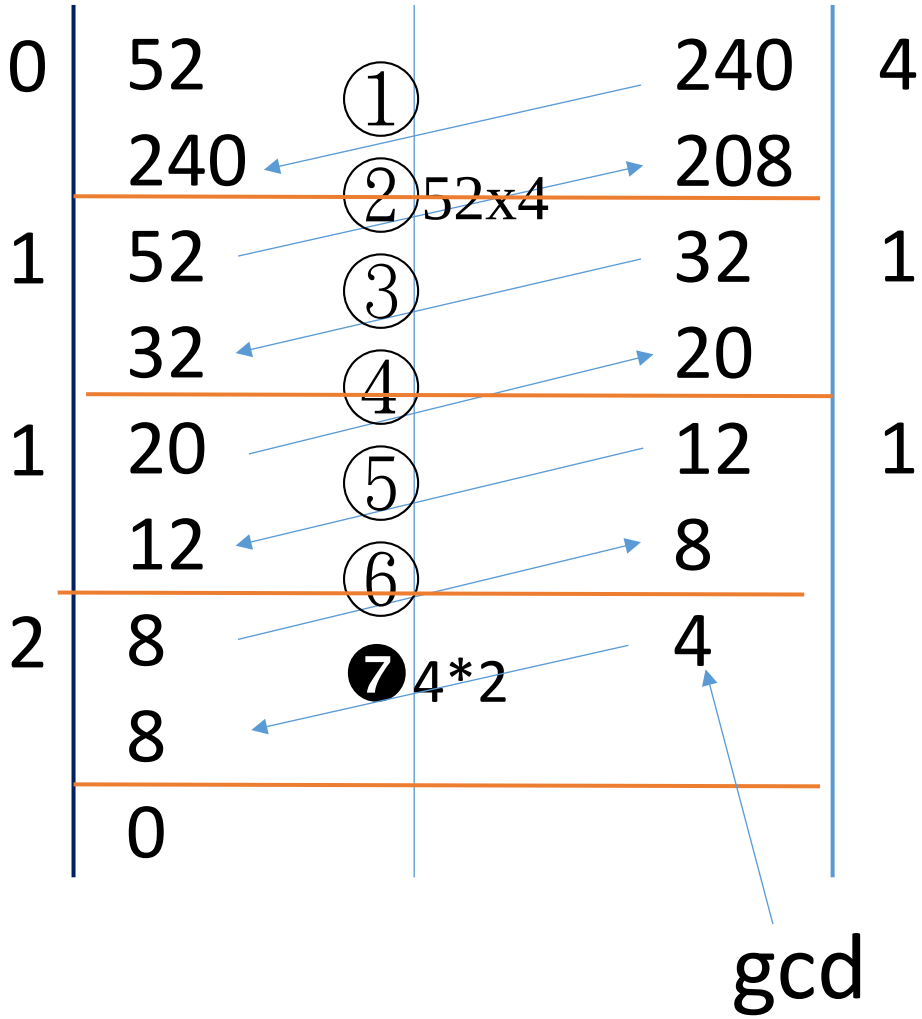
52/240=0 ……52
240/52=4……32

求兩個整數GCD的方法2: **輾轉相除法**

- **輾轉相除法**，又稱**歐幾里得算法**（Euclidean algorithm）
- 兩個整數的最大公因數(greatest common divisor)是能夠同時整除它們的最大的正整數

```
int t,m,n;
   m=n1; n=n2;
   do {
       t=m%n;
      m=n; //除數變被除數
      n=t;   //餘數變除數
      } while (n!=0); //除數不可為0
    gcd=m;
   System.out.println("輾轉相除法: GCD("+n1+","+n2+")="+gcd);
   System.out.println("輾轉相除法: LCM("+n1+","+n2+")="+n1*n2/gcd);
```

求GCD(52,240)



輾轉相除法

步驟	M (被除數)	N (除數)	t=m%n (取餘數)	m=n(除數變成被除數)	n=t(餘數變成除數)	條件 (n!=0)
(1)	52	240	52	n=240	n=52	true
(2)	240	52	32	52	32	true
(3)	52	32	20	32	20	true
(4)	32	20	12	20	12	true
(5)	20	12	8	12	8	true
(6)	12	8	4	8	4	true
(7)	8	4	t=8%4 ==0	m=4	n=0	False 結束loop
				GCD=4		

```
do {
      t=m%n; m=n; n=t;
      } while (n!=0);
  gcd=m;
```

求兩個整數GCD的方法3: 輾轉相減法

```
m=n1; n=n2;
while (m!=n) {
      while (m>n)  m=m-n;
      while (m<n)  n=n-m;  }
gcd=m; // gcd=n;
```

GCD=4

步驟	m (被減數)	n (減數)	m=m-n	n(減數不變)	條件	條件 (m!=n)
(1)	240	52	188	52	m>n	true
(2)	188	52	136	52	m>n	true
(3)	136	52	84	52	m>n	true
(4)	84	52	32	52	m>n	true
	n (被減數)	m (減數)	n=n-m	m(減數不變)		
(5)	52	32	20	32	m<n	true
	m	n	m=m-n	n		
(6)	32	20	12	32	m<n	true
	n	m	n=n-m	m		
(7)	32	12	20	12	m>n	true
(8)	20	12	8	12	m>n	true
	m	n	m=m-n	n		
(9)	12	8	4	8	m<n	true
	n	m	n=n-m	m		
(10)	8	4	4	4	m<n	False

求兩個整數GCD的方法3: 輾轉相減法

```
m=n1; n=n2;
while (m!=n) {
        while (m>n)  m=m-n;
        while (m<n)  n=n-m;  }
gcd=m;
System.out.println("輾轉相減法: GCD("+n1+","+n2+")="+gcd);
System.out.println("輾轉相減法:
LCM("+n1+","+n2+")="+n1*n2/gcd+"\n");
```

檢討GCD方法

- 求兩個整數GCD的方法:
  - 從2開始找， 直到能整除兩個整數的最大正整數
    - 2~min(n1,n2)==min(n1-1, n2-1)
  - 輾轉相除法
  - 輾轉相減法
- 那些較好?那些暴力法(brute force)?
  - 速度快

Demo: GCD的方法2

字元金字塔
Nested for loop

主題：字元金字塔 - 斜金字塔

- 利用迴圈印出「*」，逐行增加印出個數，直到印出7層斜金字塔。
- 本題利用到巢狀迴圈的概念
- 巢狀迴圈為迴圈範圍內又有迴圈，從外層來看，內層迴圈只屬與外層迴圈內的動作。因此外層迴作用，內層迴圈開始運作到執行結束後，又回到外層迴圈。

```java
public class Charstar1 {
    public static void main(String[] args) {
        //變數level為金字塔的層數
        int level = 7;

        //for迴圈 直到印完 level行結束金字塔
        //外圍迴圈i為當下的層數，i增加即為換層
        for (int i = 1; i <= level; i++) {
            //第i列時，印出i個*
            //j為當下的星星個數，每列都從1個開始印
            for (int j = 1; j <= i; j++)
                System.out.print("*");

            //每層結束換行
            System.out.println("");
        }
    }
}
```

執行結果

```
*
**
***
****
*****
******
*******
```

主題：字元金字塔 - 斜金字塔

```java
public class Charstar1 {
    public static void main(String[] args) {
        //變數level為金字塔的層數
        int level = 7;

        //for迴圈 直到印完 level行結束金字塔
        //外圍迴圈i為當下的層數，i增加即為換層
        for (int i = 1; i <= level; i++) {
            //第i列時，印出i個*
            //j為當下的星星個數，每列都從1個開始印
            for (int j = 1; j <= i; j++)
                System.out.print("*");

            //每層結束換行
            System.out.println("");
        }
    }
}
```

執行結果

```
*
**
***
****
*****
******
*******
```

追蹤nested for loop程式

外層 i	內層 j			
	初值	條件 j<=i	執行次數	結果
i=1	j=1	j<=1	內迴圈執行1次	印1顆*、換行
i=2	j=1	j<=2	內迴圈執行2次	印2顆*、換行
i=3	j=1	j<=3	內迴圈執行3次	印3顆*、換行
i=4	j=1	j<=4	內迴圈執行4次	印4顆*、換行
i=5	j=1	j<=5	內迴圈執行5次	印5顆*、換行
i=6	j=1	j<=6	內迴圈執行6次	印6顆*、換行
i=7	j=1	j<=7	內迴圈執行7次	印7顆*、換行
i=8,外迴圈結束				

輸入n層，印出n層三角形 (3層loop)

```
輸入level數:5
*
**
***
****
*****
輸入level數:4
*
**
***
****
輸入level數:3
*
**
***
輸入level數:2
*
**
```

```
import java.util.Scanner;
class Charstar1 {
 public static void main(String[] args) {
   //變數level為金字塔的層數
  Scanner input = new Scanner(System.in);
  int level=1;
  while (level>=1) {
   System.out.print("輸入level數:");
   level = input.nextInt();
   if (level<1) break;
                //for迴圈 直到印完 level行結束金字塔
                //外圍迴圈i為當下的層數，i增加即為換層
                for (int i = 1; i <= level; i++) {
                        //第i列時，印出i個*
                        //j為當下的星星個數，每列都從1個開始印
                        for (int j = 1; j <= i; j++)
                                System.out.print("*");

                        //每層結束換行
                        System.out.println("");
                }//outer for
      }//while
         }//main
}//class
```

第11周習題:

- 輸入n1, n2, n3，求三整數之GCD
- 須能重複執行
- 繳交”設計歷程”檔及.java