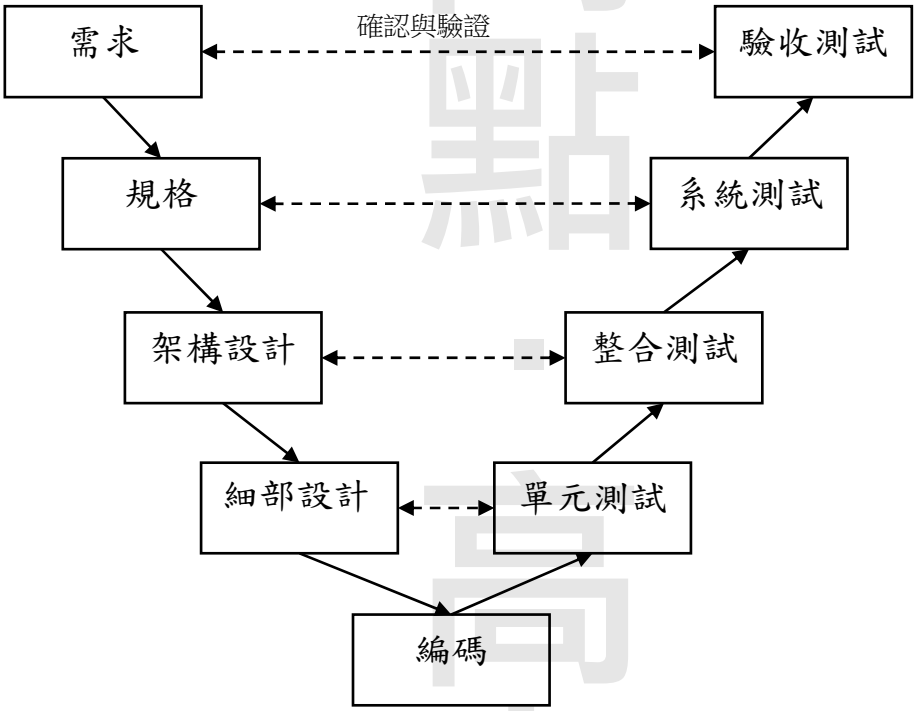
《系統專案管理》

一、V模型（The V-Model）為常見的系統開發模型之一，請繪圖並說明其特性，並從系統分析師的角度來探討其優、缺點。（25分）

試題評析	本題為系統開發方法，可由瀑布模式與軟體測試之結合切入。
考點命中	1.《高點系統專案管理講義》第二回，張又中編撰，頁2-4～2-5。 2.《高點系統專案管理講義》第四回，張又中編撰，頁4-29～30。

答：



為一般大型軟體專案常用的開發流程，其自瀑布模式改良並展現了不同抽象層次的開發與相關測試之間的關係。抽象層次可分為分析使用者需求的高階抽象層次，將需求轉為軟體架構的中階抽象層次，以及系統功能的細部設計與實作的低階抽象層次。

V模型	
優點	缺點
✚ 良好的結構化方法，每個開發階段可根據前一開發階段的產出來進行。 ✚ 可透過各階段的修正與回饋來提升軟體品質。 ✚ 可及早規劃測試計畫，節省專案時間。	✚ 專案後期需求與設計的經常變動會影響軟體品質。 ✚ 愈高層次的開發產出，其瑕疵愈晚才會被驗證出來。 ✚ 愈高層次的開發產出瑕疵將會在較低層次的開發產出中擴散與蔓延。

二、目前國內外通常採用軟體能力成熟度模式整合（Capability Maturity Model Integration，以下簡稱CMMI）或是ISO 9000以為企業本身產品（或軟體）開發能力評估與品管標準。而六個標準差（Six Sigma）則是目前工業界盛行的一種品管檢測方式，請說明CMMI與ISO 9000之異同

點。另請探討並繪圖說明CMMI階段式表述（Staged Representation）與六個標準差之間的關係。（25分）

試題評析	本題為專案管理，可由講義CMMI、ISO-9000內容作答。
考點命中	《高點系統專案管理講義》第九回，張又中編撰，頁9-26～30。

答：

CMMI為美國卡內基美隆大學(Carnegie Mellon University)的軟體工程學院(Software Engineering Institute, SEI)所發展，其將個別的模式整合成為一個完整的能力成熟度整合模式。

ISO 9000是國際標準組織設立的品質標準，例如：ISO 9000-3係針對軟體發展的過程、軟體供給與維護，規定一個最低軟體品質的標準，適合於軟體產業的應用。

標準	CMMI	ISO 9000
相同	✚ 流程導向 ✚ 持續改善	
相異	✚ 為組織訂定一套組織發展與成熟的目標與途徑。 ✚ 對於每一個流程有詳盡明確的要求。 ✚ 有階段式提升的規劃。 ✚ 各階段聚焦於數個流程。 ✚ 強調落實與產生績效。	✚ 組織自行定義品質目標以及達成目標的程序與做法。 ✚ 為一系列的標準規範，協助組織確保軟體品質。 ✚ 為通盤性的稽核標準，提供組織應執行的方向。 ✚ 強調稽核與驗證。

CMMI階段式表述採用成熟度等級(Maturity Level)來衡量企業整體的流程改善績效，強調軟體流程改善各層級的完成都是下個層級的基礎。其將組織的軟體開發能力水準分為五個成熟度階段。

六個標準差透過確認、消除引起殘疵的流程來提高產品品質，降低生產和流程中的變異，其利用一套品質管理方法，包括統計方法，根據具體的步驟以實現目標。

兩者關係如下圖所示：

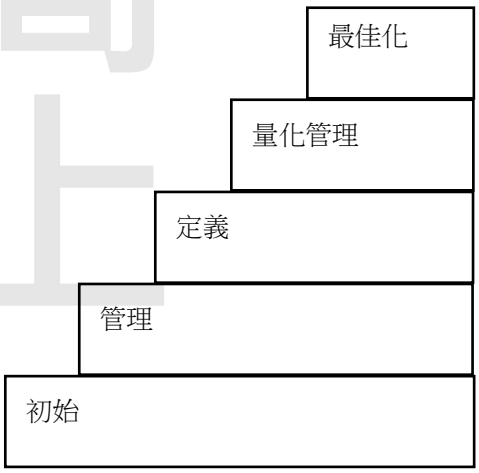
六個標準差用於全組織、關注產品與流程、聚焦於品質關鍵因子：

- ✚ 全組織的六個標準差改善與控制
- ✚ 流程領域與六個標準差關聯
- ✚ 六個標準差用於 CMMI 成果
- ✚ 六個標準差的鑽研驅動局部改善
- ✚ 適當的基礎設施
- ✚ 定義流程符合六個標準差

六個標準差的鑽研驅動局部改善

- ✚ 六個標準差的哲學與方法聚焦
- ✚ 六個標準差的鑽研驅動局部改善

六個標準差驅動、加速CMMI解決方案



三、在資訊系統開發過程中，專案管理者常會因應客戶端的要求而被迫縮短開發時間而將軟體提前釋放，而此種狀況為開發人員及專案管理者所經常面臨到的嚴峻問題與挑戰。根據國內／外研

究報告指出資訊系統開發時程的壓縮有其極限性，事實上管理者通常無法任意藉由增加開發人員與添購更多的軟、硬體設備來達到時程壓縮的目的。Putnam提出了軟體方程式（Software Equation），其定義為：$E=[LOC\times B^{0.333}/P]^3\times(1/t^4)$，其中 E 為開發心力（Development Effort，單位為人月或人年）、t 為專案執行時間、B 為特別技能因子、P 為生產力參數、LOC 為軟體大小（單位為程式碼行數）。請透過軟體方程式來舉例說明開發時程壓縮，將對開發心力造成何種程度的影響。另從實務面來看，合理且可行的時程壓縮極限應為多少？請敘述其可能原因為何？（20分）

試題評析	本題為專案管理，可由題目所給之軟體方程式以及講義例題解答進行論述。
考點命中	《高點系統專案管理講義》第十回，張又中編撰，頁10-50。

答：

根據Putnam的軟體方程式，開發心力與專案執行時間呈4次方反比，故當開發時程壓縮時，會造成開發心力的急遽增加。例如：當專案執行時間壓縮為原規劃的1/2，則開發心力將為原來的16倍。

Norden提出時間壓縮函數：$C=ht/k_2(k_1h-t)$

C為專案可回復的落後進度；h為原規劃的專案時程；t是專案剩餘時間；k_1、k_2為參數，$1.5\leqq k_1\leqq 3$，$2\leqq k_2\leqq 10$。根據其研究，專案的時程壓縮並非毫無限制，其剩餘時間有著極大的關係，專案開發前期的壓縮效果比後期好，而專案延遲的程度越低，則壓縮效果越佳。Bohem(1985)認為時程壓縮不要低於原規劃專案時程的75%，因為可能造成開發團隊的錯誤率增加而影響專案品質，甚至導致專案失敗，且過度的壓縮無助於提早完成專案。

四、針對下列八支程式模組：

（一）請完成下列表格並明確指出這些程式各具有何種內聚力（Cohesion）及說明其原因？在此內聚力型態（Cohesion Types）須從最差（Worst）至最佳（Best）依序正確排列。另說明欄中若無任何具體說明或解釋逕以零分計算。（18分）

內聚力型態	所對應之程式模組 （請以P1, P2, …等標示）	說明
⋮	⋮	⋮

（二）請針對該表格中最差（即Worst）內聚力型態之程式模組提出具體改進方法。（6分）

（三）假設吾人定義內聚力比率（Cohesion Ratio）公式如下，請據此計算出該批程式模組之內聚力比率。（6分）

$$\text{Cohesion Ratio}=\frac{\text{Number of program modules having functional cohesion}}{\text{Total number of program modules}}$$

```
//P1
public class P1 {

    public void count1(int m, int n, int p)
    {
        int counter1, counter2, counter3;
        counter1 = 1;
        cusum = 0;
        while (counter1 <= m)
        {
            cusum += counter1;
            counter1 += 1;
        }
        counter2 = 1;
        product = 1;
        while (counter2 <= n)
        {
            product *= counter2;
            counter2 += 1;
        }
        counter3 = 1;
        sum = 0;
        while (counter3 <= p)
        {
            sum += counter3;
            counter3 += 1;
        }
        mean = sum / p;
    }
    public int getSum()
    {
        return sum;
    }
    public int getProduct()
    {
        return product;
    }
    public int getCusum()
    {
        return cusum;
    }
    public int getMean()
    {
        return mean;
    }
    private int sum, product, cusum, mean;
}
```

```
//P2
public class P2 {

    public void count2(int n)
    {
        int counter;
        counter = 1;
        cusum = 0;
        product = 1;
        while (counter <= n)
        {
            cusum += counter;
            product *= counter;
            counter += 1;
        }
    }
    public int getCusum()
    {
        return cusum;
    }
    public int getProduct()
    {
        return product;
    }
    private int cusum, product;
}
```

```
//P3
public class P3 {

    public void count3(int n)
    {
        int counter;
        counter = 1;
        cusum = 0;
        while (counter <= n)
        {
            cusum += counter;
            counter += 1;
        }

        mean = cusum / n;
    }
    public int getCusum()
    {
        return cusum;
    }
    public int getMean()
    {
        return mean;
    }
    private int cusum, mean;
}
```

```
//P4
public class P4 {

	public void count4(int n)
	{
		int counter;
		counter = 1;
		cusum = 0;
		while (counter <= n)
		{
			cusum += counter;
			counter += 1;
		}
	}
	public int getCusum()
	{
		return cusum;
	}
	private int cusum;
}
```

```
//P5
public class P5 {

	public void count5(int first,int second)
	{
		int intermediate;
		intermediate = first;
		result_first = second;
		result_second = intermediate;
	}

	public int getResult_first()
	{
		return result_first;
	}
	public int getResult_second()
	{
		return result_second;
	}
	private int result_first, result_second;
}
```

```
//P6
public class P6 {

	public void count6(int n,int product)
	{
		int counter1, counter2, counter3, counter4;

		counter1 = 1;
		int a[] = new int[n];
		while (counter1 <= n)
		{
			a[counter1-1] = counter1;
			counter1 += 1;
		}
		counter2 = 0;
		cusum = 0;
		while (counter2 < n)
		{
			cusum += a[counter2];
			counter2 += 1;
		}
		counter3 = 0;
		prod = 1;
		while (counter3 < n)
		{
			prod = prod* product * a[counter3];
			counter3 += 1;
		}
		counter4 = 0;
		sum = 0;
		while (counter4 < n)
		{
			sum += a[counter4];
			counter4 += 1;
		    }
			mean = sum / n;
		}
		public int getCusum()
		{
			return cusum;
		}
		public int getProd()
		{
			return prod;
		}
		public int getSum()
		{
			return sum;
		}
		public int getMean()
		{
			return mean;
		}
		private int cusum, prod, sum, mean;
	}
```

```
//P7
public class P7 {

    public void count7(int[] tmp, int n)
    {
        int counter1, counter2, temp;
        counter1 = 0;
        a = tmp;
        System.out.print("\n");
        for (counter1 = 1; counter1 < n; counter1++)
        {
            for (counter2= 0; counter2 < counter1; counter2++)
            {
                if (a[counter1] <a[counter2])
                {
                    temp = a[counter1];
                    a[counter1] = a[counter2];
                    a[counter2] = temp;
                }
            }
        }
    }
    public int[] geta()
    {
        return a;
    }
    private int[] a;
}
```

```
//P8
public class P8 {

    public void count8(int m, int n, int p, int flag)
    {
        int counter1, counter2, counter3;
        cusum = 0;
        product = 1;
        sum = 0;
        mean = 0;
        if (flag == 1)
        {
            counter1 = 1;
            cusum = 0;
            while (counter1 <= m)
            {
                cusum +=   counter1;
                counter1 += 1;
            }
        }
        else if (flag == 2)
        {
            counter2 = 1;
            product = 1;
            while (counter2 <= n)
            {
                product *= counter2;
                counter2 += 1;
            }


        }
        else
        {
            counter3 = 1;
            sum = 0;
            while (counter3 <= p)
            {
                sum += counter3;
                counter3 += 1;
            }
        }
        mean = sum / p;
    }
    public int getCusum()
    {
        return cusum;
    }
    public int getProduct()
    {
        return product;
    }
    public int getSum()
    {
        return sum;
    }
    public int getMean()
    {
        return mean;
    }
    private int cusum, product, sum, mean;
}
```

試題評析	本題為結構化分析與設計，可由內聚力之定義切入。
考點命中	《高點系統專案管理講義》第四回，張又中編撰，頁4-19～21。

答：

(一)

內聚力型態	所對應之程式模組	說明
偶發 Coincidental	P1	模組內具多個功能，且功能間沒有關係。
邏輯 Logical	P8	模組內具多個邏輯上相關聯的功能。
暫時內聚力 Temporal	P6	模組內具多個功能，然其僅具時序關聯。
程序 Procedural	P2、P5	模組內具多個功能，其按照一定的順序執行但不共用資料。
溝通 Communication	P3	模組內具多個功能且使用相同資料，然其執行順序無相關性。
順序 Sequential	P4、P7	模組內具多個功能，且一功能的輸出立即成為下一功能的輸入。
功能 Functional	無	一模組內僅具唯一功能。

(二)在P1中，count1內部各功能彼此之間皆無關聯，建議可獨立形成新功能。例如：cusum的計算可與getSum()結合；getProduct的計算可與getProduct()結合；sum與mean之計算可與getSum()、getMean()結合。

(三)P1內聚力比率=4/5
P2內聚力比率=2/3
P3內聚力比率=2/3
P4內聚力比率=1/2
P5內聚力比率=2/3
P6內聚力比率=4/5
P7內聚力比率=1/2
P8內聚力比率=4/5