臺灣二００六年國際科學展覽會

科　　別：化學科

作 品 名 稱：化學中的數學與程式設計

得 獎 獎 項：第三名

學校 / 作者　：高雄市立高雄高級中學　蔡政江
　　高雄市立高雄高級中學　高承詣

蔡政江，1988 年 10 月 17 日生，畢業於高雄市立陽明國小及私立立志中學，現就讀高雄中學三年級，曾參加 2004 年國際數學奧林匹亞及 2005 年國際數學奧林匹亞。於校內有參加演辯社，數研社，電研社，並擔任後兩者之教學工作。目前計畫畢業後將赴美國留學主修數學。

高承詣(圖右一)，1988 年 2 月 20 日出生於高雄市，畢業於市立十全國小及市立五福國中，現就讀高雄中學三年級，曾參加清華大學第二屆全國高級中學化學科能力競賽及教育部 94 學年度高級中學化學科能力競賽。於校內有參加科學研習社化學組，目前將參加國際化學奧林匹亞選拔訓練營。

Topic：The Math and Programming in Chemistry

Abstract

When we were learning about organic compounds at school ,there was a unit discussing the isomers of alkane .Our teacher made us practice drawing all the structural formula of the isomers from hexane to nonane .We were much interested in the subject .However ,we often missed or duplicated some isomers .Thus , we began to think if it is possible to find a way by developing programs to let the computer calculate the exact number of the isomers of alkane .

After discussion ,we set up a complete coding system .We numbered the isomers in the way that computers could decode and then wrote them in C language. Through computer execution ,the numbers of the isomers from C_1 to C_{20} all match those on the reference website. According to the same concept , we also find a way to calculate the number of alkane with one substituted group . In the future,our goal will be focused on the research of multi- substituted alkane and cyclokane.

In addition , the ionic crystal accumulation model are so variable. Take the double face-centered accumulation of NaCl for example, when the ion pairs are extended to the infinity , the potential energy of attractive field will approach a constant which is named as the **Madelung Constant**. We also managed to write a computer program with C language to approach this convergence with three models, including cube , octahedron , and sphere . The result turned out to be that the data of the sphere was less stable . In the other two models , when “n” is up to 43 layers , the data is identical with that on the reference website to the eight decimal point .

中文摘要

化學中的數學&程式設計

在學校裡學習有機化合物有關烷類異構物這個單元，老師讓我們練習畫出己烷~壬烷的所有異構物結構式，這引起我們極大的興趣！但常一不小心就漏掉或多出幾個，我們開始思考：能不能找到一個方法並設計成程式，讓電腦執行以找出烷類異構物？

經過討論，我們建立了一套完整的編碼系統，將各異構物以電腦可解讀的方式編號，並以 C 語言寫成程式。透過電腦執行，各碳數化合物自 C_1 至 C_{20} 都與參考網站吻合。依相同觀念，我們也設計出烷類含一個取代基的異構物數目。將來努力的目標為：多取代基及環烷類之研究！

另外，離子晶體堆積模型變化多端，以 NaCl 雙面心堆積為例，其引力場位能，當離子對延伸至無限大時，這個值將趨近於一個常數，又稱為**馬德隆常數**。我們嘗試以 C 語言設計電腦程式，用三種模型（正立方體、正八面體、圓球）來逼近並求得這個收斂值。執行結果是：圓球數據較不穩定；而另二種模型到 n=43 層以上，其數值大小與參考網路上的數值，在小數點以下 8 位完全相同。

一、前言

（一）研究動機

1. 學校化學老師教到「化學鍵與烴類」時，說明了烷、烯、炔的定義、結構、命名，並向我們解釋何謂同分異構物，接著就以鏈狀烷類爲例：甲烷、乙烷、丙烷都只有 1 種，丁烷有 2 種，戊烷有 3 種，己烷有 5 種，庚烷有 9 種，辛烷有 18 種，壬烷有 35 種，癸烷有 75 種。說到這，老師給了我們一項寒假作業：畫出所有 75 種癸烷異構物。我們幾人迫不及待投入其中，只是一開始畫總是不小心漏或多幾個。好不容易完成後，我們就想：人總是很容易粗心，才 10 個碳就會出錯，能不能找出一套完整的方法並設計成程式，讓電腦執行找出其異構物？

2.上課的時候學到物質的結構時，對離子晶體的各種堆積模型和方式特別感興趣。原來離子化合物的晶體堆積竟是如此變化多端！
在學離子晶體的引力場位能時，形成離子鍵，有一個最穩定的距離，也就是此時這個離子對的位能降的最低，故也最穩定，以 NaCl 爲例，這個數值是
-492 kJ /mol。

但是我們知道其實離子晶體並不是單單一兩對離子對堆積而成的。所以在有很多離子對一起堆積的情況下，這個值跟單一對離子堆積會不會有差距呢？
關於這個問題，我們請教老師後，得知其實這個問題很早就被提出了，也有人曾以數學式子表示出雙面心的引力場位能，當離子對延伸至無限大時，這個值事實上會趨近於某個數，也就是說，把整個式子展開後，它將會是一個收斂級數，又稱爲**馬德隆常數**。

我們以雙面心堆積的 NaCl 做探討模型。發覺其實馬德隆常數和實際測得 NaCl 的格子能是有差距的。

由於有電腦的輔助，我們曾做過以正立方體的型式來求這個收斂值。但我們又知道其實雙面心對每一種離子來說都可以看成是一個正八面體的模型。
而離子晶體無限延伸後，也將會更趨近一個圓球。

所以我們嘗試以 **1.正立方體 2.正八面體 3.圓球** 的模型來逼近並求得這個收斂值，求出來的收斂值會不會有什麼不一樣呢？

（二）研究目的

1、設計一套電腦程式來找出有機烷類的異構物。
2、設計一套電腦程式來求馬德隆常數。

二、研究方法或過程

（一）研究方法

電腦、C 語言。

（二）研究過程

1. 尋找異構物相關資料

知道我們的想法後，老師提供我們一個國外網站：“Molecular Fragments: Alkane Isomers” (http://people.ouc.bc.ca/woodcock/molecule/molfrag/mf009.htm)

站中首先說明了同分異構物的定義和類型，接者列出[表一]。

▼ [表一]

碳原子數	結構異構物數
4	2
5	3
6	5
7	9
8	18
9	35
10	75
12	355
15	4,347
20	366,319

最後列出由丁烷(C_4H_{10})到癸烷($C_{10}H_{22}$)所有異構物之 IUPAC 命名並註明含光學異構物者。

但是站中未提及如何找出這些異構物，也未註明表一中碳原子數爲 15、20 之鏈狀烷異構物數目是從何得知(顯然不可能是人工算出)。對我們而言，這只能作爲參考資料，程式撰寫的基礎構想仍得靠我們自己討論。

2. 構思並初步設計程式

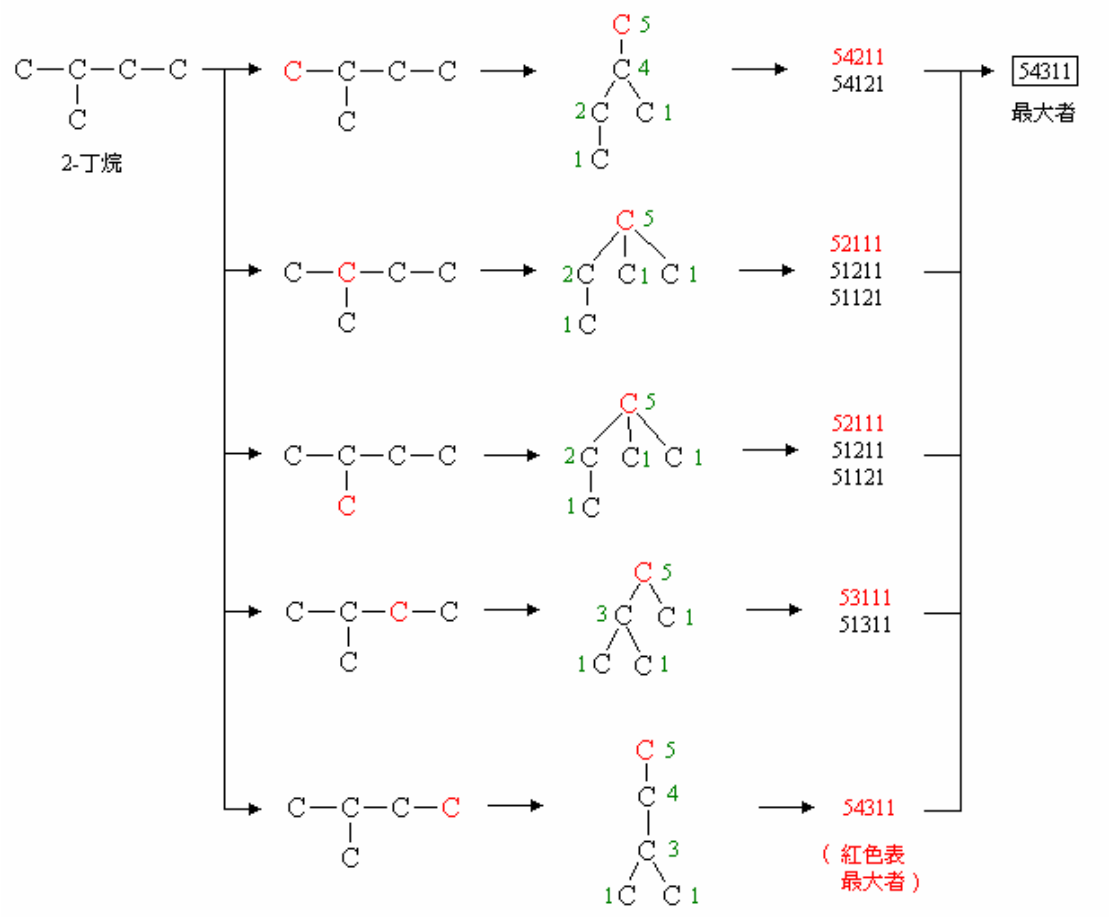
(A)我們首先討論出下列結果：

(1)必須建立一套完整編碼系統將各異構物以電腦可解讀的方式編號。

(2)由於處理的是鏈狀烷的結構異構物，編碼系統只須注意各碳原子間的鍵結關係，只要碳原子的鍵結數不超過 4，可忽略氫原子，而光學異構物不列入討論。

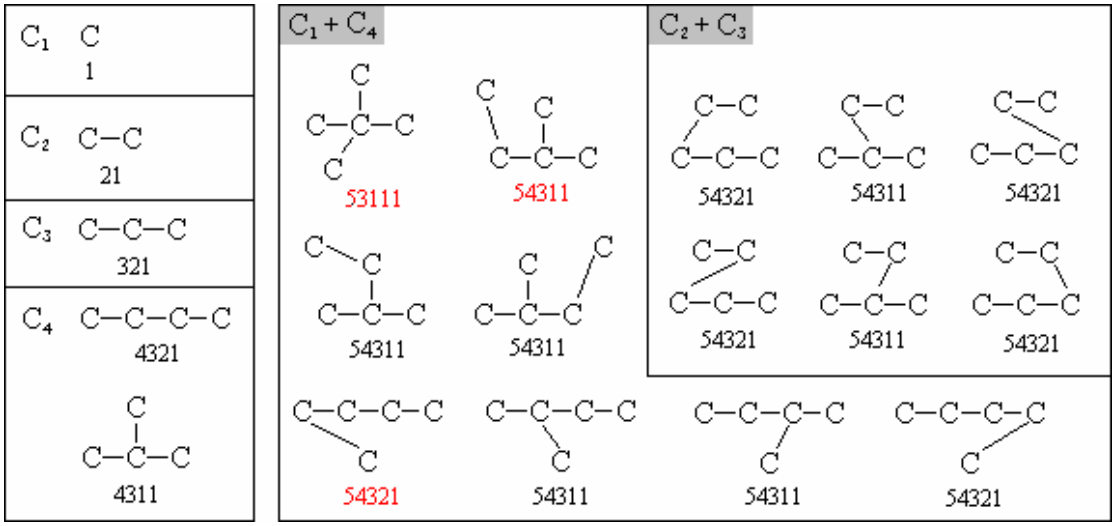
(3)若同一異構物產生兩種以上編碼，系統須按一定方法輸出其中一種，以免重複。

(B)針對某一烷類結構，以下是我們的編碼構想：(以 2-甲基丁烷爲例，如[圖一])
(1)選定其中一碳原子，把它抓起來抖一抖。
(2)算出所有碳原子以下(含此原子)的碳原子數。
(3)由上而下，依各支鏈將所有數字寫成一排(可能有多組)，選出最大者。
(4)換另一碳原子，再把它抓起來抖一抖。重複 2、3 步驟。
(5)比較求得的各數，最大者即爲其編碼，[圖一] 最右加框者。



▲ [圖一]

(C)至於程式如何找出某碳數烷類之所有異構物，構想如下：(以下碳數爲 n 之烷類以 C_n 表示)(以 C_5 爲例，如〔圖二〕)
(1)欲求 C_x，則先找出 C_k (k=1 ~ x-1)的所有異構物(其編碼)。
(2)將 C_k 之任一碳和 C_{x-k} 之任一碳接在一起後予以編碼。
(3)過濾掉重複者即可輸出。



▲ [圖二](圖中加框表輸出者)

(D)依據 A、B、C，我們利用 C 語言寫成一程式。程式碼如附件。

3.查閱*馬德隆常數*相關資料

- **離子正方**

$$E_{ion\ square} = \left[\overbrace{4 \times K\frac{(+e)(-e)}{d}}^{\text{引力能}} + \overbrace{K\frac{(+e)(+e)}{\sqrt{2}d} + K\frac{(-e)(-e)}{\sqrt{2}d}}^{\text{斥力能}} \right] \times N_0$$

$$= (\sqrt{2}-4) \times K \times \frac{e^2}{d} \times N_0$$

$$= (\sqrt{2}-4) \times 9.0 \times 10^9 \times \frac{(1.60) \times 10^{-19})^2}{2.81 \times 10^{-10}} \times 6.02 \times 10^{23}$$

$$= -2.586 \times 492\ kJ/mol$$

馬德隆常數

若 Na^+與 Cl^-形成三度空間的晶體格子時，可估計能量釋放 864KJ/mol。在三度空間 1 個 Na^+周圍有六個 Cl^-接觸產生引力場位能。12 個 Na^+在√2d 處產生斥力位能；在√3d 處有 8 個 Cl^-產生引力位能，……等等。

- **無限延伸**

$$
\begin{aligned}
E_{total} &= (-K \times \frac{e^2}{d} \times N_0) \times 6 + (K \times \frac{e^2}{\sqrt{2}d} \times N_0) \times 12 \\
&\quad + (-K \times \frac{e^2}{\sqrt{3}d} \times N_0) \times 8 + (K \times \frac{e^2}{\sqrt{4}d} \times N_0) \times 6 \\
&\quad + (-K \times \frac{e^2}{\sqrt{5}d} \times N_0) \times 24 + (K \times \frac{e^2}{\sqrt{6}d} \times N_0) \times 24 + \ldots \\
&= K \times \frac{e^2}{d} \times N_0 \times (-\frac{6}{\sqrt{1}} + \frac{12}{\sqrt{2}} - \frac{8}{\sqrt{3}} + \frac{6}{\sqrt{4}} - \frac{24}{\sqrt{5}} + \frac{24}{\sqrt{6}} - \ldots) \\
&= -492 \times (1.74) \\
&= -864\ kJ/mol
\end{aligned}
$$

其中此無限收斂級數的和即 1.74 (Madelung constant，**馬德隆常數**。)

事實上，若將上述式子以座標的方式表達，即可將其化簡爲

$$M_3 = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \frac{(-1)^{(i+j+k)}}{\sqrt{(i^2 + j^2 + k^2)}}$$

若定中心原子的座標系爲(0, 0, 0)
(i, j, k) 則表示外圍原子的座標
正負號則決定其引力或斥力。

4.實驗過程

以正立方體爲例

$$E_{total} = (-K \times \frac{e^2}{d} \times N_0) \times 6 + (K \times \frac{e^2}{\sqrt{2}d} \times N_0) \times 12$$
$$+ (-K \times \frac{e^2}{\sqrt{3}d} \times N_0) \times 8 + (K \times \frac{e^2}{\sqrt{4}d} \times N_0) \times 6$$
$$+ (-K \times \frac{e^2}{\sqrt{5}d} \times N_0) \times 24 + (K \times \frac{e^2}{\sqrt{6}d} \times N_0) \times 24 + \ldots$$

n …… **1 2 3 4 5 6 …**

$$= K \times \frac{e^2}{d} \times N_0 \times (-\frac{6}{\sqrt{1}} + \frac{12}{\sqrt{2}} - \frac{8}{\sqrt{3}} + \frac{6}{\sqrt{4}} - \frac{24}{\sqrt{5}} + \frac{24}{\sqrt{6}} - \ldots)$$

= 492*(X) …… (X 即爲 print 出來的收斂值，會隨 **n** 的不同而改變)

三、研究結果與討論

（一）有關異構物

1.程式執行

執行結果：各碳數異構物如[表二]。自 C_1 至 C_{10} 爲止皆與參考網站上之數據吻合，但之後就愈差愈多。

[表二]

碳原子數	結構異構物數	碳原子數	結構異構物數
1	1	11	161
2	1	12	362
3	1	13	820
4	2	14	1917
5	3	15	4509
6	5	16	10796
7	9	17	26054
8	18	18	63710
9	35	19	156633
10	75	20	388898

2.修正程式

我們設計的程式執行結果與參考網站資料有所出入，顯示兩者必有其一錯誤。假設參考網站是正確的，而我們的構想又沒有問題，那麼錯誤肯定出在程式設計上，而且在碳數較大時才會發生。

經過檢查，我們找出了程式錯誤的地方：(如附件一)

更正後如下：(如附件二)

3.再執行

執行結果：各碳數異構物如[表三]。自 C_1 至 C_{20} 都與參考網站吻合。

▼ [表三]

碳原子數	結構異構物數	碳原子數	結構異構物數	執行時間(秒)
1	1	11	159	<1
2	1	12	355	
3	1	13	802	
4	2	14	1858	
5	3	15	4347	1
6	5	16	10359	3
7	9	17	24894	7
8	18	18	60523	19
9	35	19	148284	57
10	75	20	366319	167

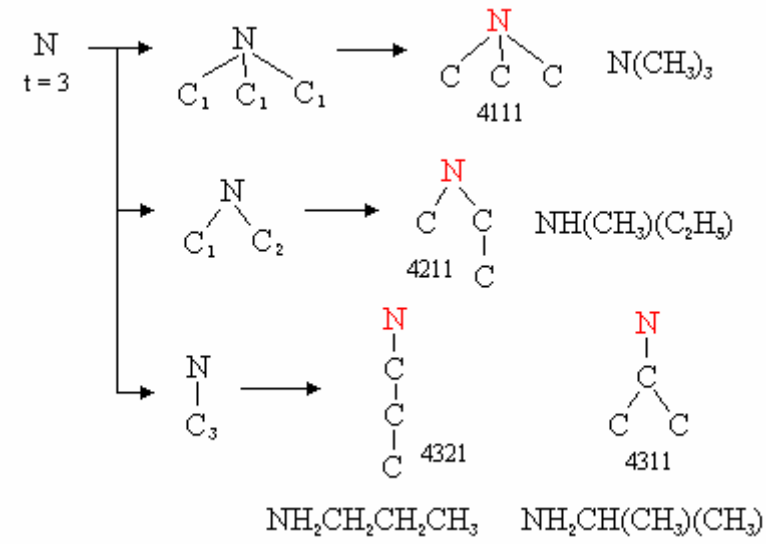
（二）有關馬德隆常數

我們發現，以正立方體來求這個收斂值，求出來的極值從一開始就十分穩定，看不出什麼明顯的變化。既不像逼近到圓球的方式，極值的大小很沒有規則，就算數字已經很大，約莫還是在 1.70-1.99 之間來回跳動；也沒有明顯的成長。如果以正八面體堆積，數據從一開始就與正立方體差距不多，到 n=43 以上，小數點下 8 位就全部相等了，這個值(1.74756459)，也和文獻中的數據非常符合〈1.747564594〉。換句話說，我們以兩種不同模型，推算出的這個收斂值，確實和馬德隆常數相當！

四、結論與應用

（一）有關異構物

1.目前我們已設計出程式列出直鏈烷類之所有異構物(如表三)。
2.按相同觀念，可將其他原子或原子團(但只能一個)取代物置入某碳數之直鏈烷類計算其異構物數：(以 C_3H_9N 爲例，如[圖三])
(1)若碳數爲 x，則先找出 C_k (k=1 ~ x)的所有異構物(其編碼)作爲烷基。
(2)抓起來欲插入之原子或原子團，若欲其鍵結量爲 t(在一般情況下)，則在其下接上 1~t 個烷基之任一碳上，但所有烷基的碳數總和須爲 x。
(3)算出所有原子或原子團以下(含此原子或原子團)的原子數。
(4)由上而下，依各支鏈將所有數字寫成一排(可能有多組)，選出最大者。
(5)比較求得的各數，最大者即爲其編碼，即可輸出。



▲ [圖三]

3.$C_{10}H_{22}$ 的各異構物（含編碼）對照表，如〔附件四〕。
4.含 N, Si, O, Cl 之烷類異構物，也可以同時由程式執行跑出答案。
5.異構物數隨碳數以約 **2~3** 倍的速率成長。
6.若碳數大於 21 則電腦會因記憶體不足，而無法在螢幕上呈現。
7.以下是程式尚不能完成的，有待研究：
(1)直鏈烷上有多個原子或原子團。
(2)環烷類。
8.本程式所佔的記憶體很小(約 8k)，若化學網站有興趣者，可提供參考！

（二）有關馬德隆常數

1.圓球形的數據是最不穩定的，最大有可能到 2.2，最小也有到 1.45 的，而且每增加 1，數據的改變幅度也很大，探討起來也比較複雜。

雖然說圓球形的數據比較不穩定***(如表四)***，事實上我們知道，NaCl 固體晶格的能量(-765KJ/mol)，實際上比 492×(1.74)，也就是馬德隆常數的理論值要來的小。在數字做到 80 以上，以圓球逼近出來的極限值，有的將更爲接近實際值(≒1.55487)。這可能是因爲，雙面心堆積延伸到無限大時，整個晶格將更趨近於圓球的結果吧！

2.至於談到誤差的部分，不僅是圓球，正八面體在一開始也常有數字不規則增減的情況發生。我們想這可能是因爲數字給的不夠大，最外一層原子的引力(或斥力)，還是對中心原子造成很大的影響，因此不可忽略的緣故。

表四

	收斂值(圓球)	收斂值(正八面體)
n=8	1.784857934725	1.747564483535
n=10	1.774481110128	1.747564565691
n=43	2.129520838521	**1.74756459**4635
n=80	1.748024028630	**1.74756459**4632
n=100	1.452919340237	**1.74756459**4632
n=800	1.681385371838	**1.74756459**3677
n=1000	1.881990626174	**1.74756459**4411
n=1200	1.834055780034	**1.74756459**5572

五、參考資料

1.Molecular Fragments: Alkane Isomers”
(http://people.ouc.bc.ca/woodcock/molecule/molfrag/mf009.htm)

2.http://pi.lacim.uqam.ca/piDATA/madelung.txt

3. http://www.mathsoft.com/article/0,,2305,00.html

附件一

Alkane Isomers　的程式設計(舊)

```
#include<stdio.h>
#include<stdlib.h>
#define NoMC 22
__int8 *rec;
int NoC=0;
bool printisomers;
int compare(const void *a, const void *b)
{
	int i;
	for(i=0;i<((int *)a)[0] || i<((int *)b)[0];i++)
	{
		if( ((int *)a)[i]<((int *)b)[i] )
			return 1;
		if( ((int *)a)[i]>((int *)b)[i] )
			return -1;
	}
	return 0;
}
struct C
{
	int degree;
	int bond[4];
};
int found=0, found1=0, found2=0, found3=0;
C tree[25];
int used,get;
void construct(int source)
{
	int noc=rec[get++]-1; // Number of children
	int root=used++;
	tree[root].degree=1;
	if(source==-1)
		tree[root].degree=0;
	else
	{
		tree[root].bond[0]=source;
		tree[source].bond[tree[source].degree]=root;
		tree[source].degree=tree[source].degree+1;
	}
	while(noc>0)
	{
		noc-=rec[get];
		construct(root);
	}
}
int de[25][25];
int de0[25];
void rname(int root);
void check(int last)
{
	int i;
	used=0;
	for(i=0;i<NoC;i++)
		de0[i]=rec[last-NoC+i];
	get=last-NoC;
	construct(-1);
	for(i=0;i<NoC;i++)
```

```
            rname(i);
        qsort(de, NoC, 25*sizeof(int), compare);
        if(compare(de0,de[0])<=0)
        {
            if(printisomers)
            {
                for(i=0;i<NoC;i++)
                    printf("%d ",de[0][i]);
                printf("\n");
            }
            found++;
            int mul=NoC, test, deg;
            de[NoC][0]=0;
            for(i=0;i<NoC;i++)
            {
                if(compare(de[i],de[i+1])==0)
                    mul--;
                else
                {
                    test=1;
                    for(deg=0;test<NoC;deg++)
                        test+=de[i][test];
                    if(deg==4)
                        found2--;
                    else if(deg==3)
                        found3--;
                }
            }
            found1+=mul;
        }
}
int runned[25],subcode[25][25];
void coding1(int root);
void rname(int root)
{
        int i;
        for(i=0;i<NoC;i++)
            runned[i]=0;
        coding1(root);
        for(i=0;i<NoC;i++)
            de[root][i]=subcode[root][i];
}
void coding1(int root)
{
        int h=0,i,j,k,n=tree[root].degree,temp;
        int source=-1;
        subcode[root][0]=1;
        runned[root]=1;
        for(i=0;i<n;i++)
        {
            if(runned[tree[root].bond[i]]==0)
            {
                coding1(tree[root].bond[i]);
                subcode[root][0]+=subcode[tree[root].bond[i]][0];
            }
            else
                source=i;
        }
        if(source>0)
```

```
{
	temp=tree[root].bond[0];
	tree[root].bond[0]=tree[root].bond[source];
	tree[root].bond[source]=temp;
}
else if(source==-1)
{
	for(i=1;i<n;i++)
	{
		if(compare(subcode[tree[root].bond[0]],subcode[tree[root].bond[i]])>0)
		{
			temp=tree[root].bond[0];
			tree[root].bond[0]=tree[root].bond[i];
			tree[root].bond[i]=temp;
		}
	}
	for(h=0;h<subcode[tree[root].bond[0]][0];h++)
		subcode[root][h+1]=subcode[tree[root].bond[0]][h];
}
if(n==2)
{
	for(i=0;i<subcode[tree[root].bond[1]][0];i++)
		subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
}
else if(n==3)
{
	if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[2]])<0)
	{
		for(i=0;i<subcode[tree[root].bond[1]][0];i++)
			subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
		for(j=0;j<subcode[tree[root].bond[2]][0];j++)
			subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
	}
	else
	{
		for(i=0;i<subcode[tree[root].bond[2]][0];i++)
			subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
		for(j=0;j<subcode[tree[root].bond[1]][0];j++)
			subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
	}
}
else if(n==4)
{
	if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[2]])<0)
	{
		if(compare(subcode[tree[root].bond[2]],subcode[tree[root].bond[3]])<0)
		{
			for(i=0;i<subcode[tree[root].bond[1]][0];i++)
				subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
			for(j=0;j<subcode[tree[root].bond[2]][0];j++)
				subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
			for(k=0;k<subcode[tree[root].bond[3]][0];k++)
				subcode[root][h+i+j+k+1]=subcode[tree[root].bond[3]][k];
		}
		else
		{
			if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[3]])<0)
			{
				for(i=0;i<subcode[tree[root].bond[1]][0];i++)
```

```
                                subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
                        for(j=0;j<subcode[tree[root].bond[3]][0];j++)
                                subcode[root][h+i+j+1]=subcode[tree[root].bond[3]][j];
                        for(k=0;k<subcode[tree[root].bond[2]][0];k++)
                                subcode[root][h+i+j+k+1]=subcode[tree[root].bond[2]][k];
                }
                else
                {
                        for(i=0;i<subcode[tree[root].bond[3]][0];i++)
                                subcode[root][h+i+1]=subcode[tree[root].bond[3]][i];
                        for(j=0;j<subcode[tree[root].bond[1]][0];j++)
                                subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
                        for(k=0;k<subcode[tree[root].bond[2]][0];k++)
                                subcode[root][h+i+j+k+1]=subcode[tree[root].bond[2]][k];
                }
        }
    }
    else
    {
        if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[3]])<0)
        {
            for(i=0;i<subcode[tree[root].bond[2]][0];i++)
                subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
            for(j=0;j<subcode[tree[root].bond[1]][0];j++)
                subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
            for(k=0;k<subcode[tree[root].bond[3]][0];k++)
                subcode[root][h+i+j+k+1]=subcode[tree[root].bond[3]][k];
        }
        else
        {
            if(compare(subcode[tree[root].bond[2]],subcode[tree[root].bond[3]])<0)
            {
                for(i=0;i<subcode[tree[root].bond[2]][0];i++)
                    subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
                for(j=0;j<subcode[tree[root].bond[3]][0];j++)
                    subcode[root][h+i+j+1]=subcode[tree[root].bond[3]][j];
                for(k=0;k<subcode[tree[root].bond[1]][0];k++)
                    subcode[root][h+i+j+k+1]=subcode[tree[root].bond[1]][k];
            }
            else
            {
                for(i=0;i<subcode[tree[root].bond[3]][0];i++)
                    subcode[root][h+i+1]=subcode[tree[root].bond[3]][i];
                for(j=0;j<subcode[tree[root].bond[2]][0];j++)
                    subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
                for(k=0;k<subcode[tree[root].bond[1]][0];k++)
                    subcode[root][h+i+j+k+1]=subcode[tree[root].bond[1]][k];
            }
        }
    }
  }
}
void main()
{
    int i,j,k,l;
    int head[25],ptr,ptr2,ptr3,last=1;
    while(NoC<2 || NoC>NoMC)
    {
        printf("Number of C is:");
```

```
        scanf("%d",&NoC);
}
char print=0;
while( (print-'y')*(print-'Y')*(print-'n')*(print-'N')!=0 )
{
        printf("Print all isomers?(y/n)");
        scanf("\n%c",&print);
}
if( (print-'y')*(print-'Y')==0 )
        printisomers=1;
else
        printisomers=0;
int mu=132;
for(i=0;i<NoC;i++)
        mu*=2;
rec=(__int8 *)malloc(mu);
rec[0]=1;
head[1]=0;
for(i=2;i<NoC;i++)
{
        head[i]=last;
        for(ptr=head[i-1];ptr<head[i];ptr+=(i-1))
        {
                rec[last]=i;
                for(j=0;j<i-1;j++)
                        rec[last+j+1]=rec[ptr+j];
                last+=i;
        }
        for(j=i-2;3*j>i-2;j--)
        {
                for(ptr=head[j];ptr<head[j+1];ptr+=j)
                {
                        if(2*j>i-2)
                        {
                                for(ptr2=head[i-j-1];ptr2<head[i-j];ptr2+=(i-j-1))
                                {
                                        rec[last]=i;
                                        for(k=0;k<j;k++)
                                                rec[last+k+1]=rec[ptr+k];
                                        last+=j+1;
                                        for(k=0;k<i-j-1;k++)
                                                rec[last+k]=rec[ptr2+k];
                                        last+=(i-j-1);
                                }
                        }
                        k=i-j-2;
                        if(k>j)
                                k=j;
                        for(;2*k>i-j-2;k--)
                        {
                                for(ptr2=head[k];ptr2<head[k+1];ptr2+=k)
                                {
                                        for(ptr3=head[i-j-k-1];ptr3<head[i-j-k];ptr3+=(i-j-k-1))
                                        {
                                                rec[last]=i;
                                                for(l=0;l<j;l++)
                                                        rec[last+l+1]=rec[ptr+l];
                                                last+=j+1;
                                                for(l=0;l<k;l++)
```

```
                                        rec[last+l]=rec[ptr2+l];
                                last+=k;
                                for(l=0;l<i-j-k-1;l++)
                                        rec[last+l]=rec[ptr3+l];
                                last+=(i-j-k-1);
                        }
                }
        }
    }
    head[NoC]=last;
    for(ptr=head[NoC-1];ptr<head[NoC];ptr+=(NoC-1))
    {
        rec[last]=NoC;
        for(j=0;j<NoC-1;j++)
            rec[last+j+1]=rec[ptr+j];
        check(last+NoC);
    }
    found2+=found1;
    found3+=found2;
    printf("%d isomers of C%dH%d found.\n",found,NoC,2*NoC+2);
    printf("%d isomers of C%dH%dSi found.\n",found1,NoC-1,2*NoC+2);
    printf("%d isomers of C%dH%dN found.\n",found2,NoC-1,2*NoC+1);
    printf("%d isomers of C%dH%dCl found.\n",found2,NoC,2*NoC+1);
    printf("%d isomers of C%dH%dO found.\n",found3,NoC-1,2*NoC);
    printf("Memory used:%d bytes.\n",last);
    system("pause");
}
```

附件二

Alkane Isomers　含單一原子取代基

```
#include<stdio.h>
#include<stdlib.h>
#define NoMC 22
__int8 *rec;
int NoC=0;
bool printisomers;
int compare(const void *a, const void *b)
{
	int i;
	for(i=0;i<((int *)a)[0] || i<((int *)b)[0];i++)
	{
		if( ((int *)a)[i]<((int *)b)[i] )
			return 1;
		if( ((int *)a)[i]>((int *)b)[i] )
			return -1;
	}
	return 0;
}
struct C
{
	int degree;
	int bond[4];
};
int found=0, found1=0, found2=0, found3=0;
C tree[25];
int used,get;
void construct(int source)
{
	int noc=rec[get++]-1; // Number of children
	int root=used++;
	tree[root].degree=1;
	if(source==-1)
		tree[root].degree=0;
	else
	{
		tree[root].bond[0]=source;
		tree[source].bond[tree[source].degree]=root;
		tree[source].degree=tree[source].degree+1;
	}
	while(noc>0)
	{
		noc-=rec[get];
		construct(root);
	}
}
int de[25][25];
int de0[25];
void rname(int root);
void check(int last)
{
	int i;
	used=0;
	for(i=0;i<NoC;i++)
		de0[i]=rec[last-NoC+i];
	get=last-NoC;
	construct(-1);
	for(i=0;i<NoC;i++)
		rname(i);
```

```
	qsort(de, NoC, 25*sizeof(int), compare);
	if(compare(de0,de[0])<=0)
	{
		if(printisomers)
		{
			for(i=0;i<NoC;i++)
				printf("%d ",de[0][i]);
			printf("\n");
		}
		found++;
		int mul=NoC, test, deg;
		de[NoC][0]=0;
		for(i=0;i<NoC;i++)
		{
			if(compare(de[i],de[i+1])==0)
				mul--;
			else
			{
				test=1;
				for(deg=0;test<NoC;deg++)
					test+=de[i][test];
				if(deg==4)
					found2--;
				else if(deg==3)
					found3--;
			}
		}
		found1+=mul;
	}
}
int runned[25],subcode[25][25];
void coding1(int root);
void rname(int root)
{
	int i;
	for(i=0;i<NoC;i++)
		runned[i]=0;
	coding1(root);
	for(i=0;i<NoC;i++)
		de[root][i]=subcode[root][i];
}
void coding1(int root)
{
	int h=0,i,j,k,n=tree[root].degree,temp;
	int source=-1;
	subcode[root][0]=1;
	runned[root]=1;
	for(i=0;i<n;i++)
	{
		if(runned[tree[root].bond[i]]==0)
		{
			coding1(tree[root].bond[i]);
			subcode[root][0]+=subcode[tree[root].bond[i]][0];
		}
		else
			source=i;
	}
	if(source>0)
	{
```

```
        temp=tree[root].bond[0];
        tree[root].bond[0]=tree[root].bond[source];
        tree[root].bond[source]=temp;
}
else if(source==-1)
{
        for(i=1;i<n;i++)
        {
                if(compare(subcode[tree[root].bond[0]],subcode[tree[root].bond[i]])>0)
                {
                        temp=tree[root].bond[0];
                        tree[root].bond[0]=tree[root].bond[i];
                        tree[root].bond[i]=temp;
                }
        }
        for(h=0;h<subcode[tree[root].bond[0]][0];h++)
                subcode[root][h+1]=subcode[tree[root].bond[0]][h];
}
if(n==2)
{
        for(i=0;i<subcode[tree[root].bond[1]][0];i++)
                subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
}
else if(n==3)
{
        if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[2]])<0)
        {
                for(i=0;i<subcode[tree[root].bond[1]][0];i++)
                        subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
                for(j=0;j<subcode[tree[root].bond[2]][0];j++)
                        subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
        }
        else
        {
                for(i=0;i<subcode[tree[root].bond[2]][0];i++)
                        subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
                for(j=0;j<subcode[tree[root].bond[1]][0];j++)
                        subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
        }
}
else if(n==4)
{
        if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[2]])<0)
        {
                if(compare(subcode[tree[root].bond[2]],subcode[tree[root].bond[3]])<0)
                {
                        for(i=0;i<subcode[tree[root].bond[1]][0];i++)
                                subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
                        for(j=0;j<subcode[tree[root].bond[2]][0];j++)
                                subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
                        for(k=0;k<subcode[tree[root].bond[3]][0];k++)
                                subcode[root][h+i+j+k+1]=subcode[tree[root].bond[3]][k];
                }
                else
                {
                        if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[3]])<0)
                        {
                                for(i=0;i<subcode[tree[root].bond[1]][0];i++)
                                        subcode[root][h+i+1]=subcode[tree[root].bond[1]][i];
```

```
                        for(j=0;j<subcode[tree[root].bond[3]][0];j++)
                            subcode[root][h+i+j+1]=subcode[tree[root].bond[3]][j];
                        for(k=0;k<subcode[tree[root].bond[2]][0];k++)
                            subcode[root][h+i+j+k+1]=subcode[tree[root].bond[2]][k];
                    }
                    else
                    {
                        for(i=0;i<subcode[tree[root].bond[3]][0];i++)
                            subcode[root][h+i+1]=subcode[tree[root].bond[3]][i];
                        for(j=0;j<subcode[tree[root].bond[1]][0];j++)
                            subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
                        for(k=0;k<subcode[tree[root].bond[2]][0];k++)
                            subcode[root][h+i+j+k+1]=subcode[tree[root].bond[2]][k];
                    }
                }
            }
            else
            {
                if(compare(subcode[tree[root].bond[1]],subcode[tree[root].bond[3]])<0)
                {
                    for(i=0;i<subcode[tree[root].bond[2]][0];i++)
                        subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
                    for(j=0;j<subcode[tree[root].bond[1]][0];j++)
                        subcode[root][h+i+j+1]=subcode[tree[root].bond[1]][j];
                    for(k=0;k<subcode[tree[root].bond[3]][0];k++)
                        subcode[root][h+i+j+k+1]=subcode[tree[root].bond[3]][k];
                }
                else
                {
                    if(compare(subcode[tree[root].bond[2]],subcode[tree[root].bond[3]])<0)
                    {
                        for(i=0;i<subcode[tree[root].bond[2]][0];i++)
                            subcode[root][h+i+1]=subcode[tree[root].bond[2]][i];
                        for(j=0;j<subcode[tree[root].bond[3]][0];j++)
                            subcode[root][h+i+j+1]=subcode[tree[root].bond[3]][j];
                        for(k=0;k<subcode[tree[root].bond[1]][0];k++)
                            subcode[root][h+i+j+k+1]=subcode[tree[root].bond[1]][k];
                    }
                    else
                    {
                        for(i=0;i<subcode[tree[root].bond[3]][0];i++)
                            subcode[root][h+i+1]=subcode[tree[root].bond[3]][i];
                        for(j=0;j<subcode[tree[root].bond[2]][0];j++)
                            subcode[root][h+i+j+1]=subcode[tree[root].bond[2]][j];
                        for(k=0;k<subcode[tree[root].bond[1]][0];k++)
                            subcode[root][h+i+j+k+1]=subcode[tree[root].bond[1]][k];
                    }
                }
            }
        }
}
void main()
{
    int i,j,k,l;
    int head[25],ptr,ptr2,ptr3,last=1;
    while(NoC<2 || NoC>NoMC)
    {
        printf("Number of C is:");
        scanf("%d",&NoC);
```

```
}
char print=0;
while( (print-'y')*(print-'Y')*(print-'n')*(print-'N')!=0 )
{
	printf("Print all isomers?(y/n)");
	scanf("\n%c",&print);
}
if( (print-'y')*(print-'Y')==0 )
	printisomers=1;
else
	printisomers=0;
int mu=132;
for(i=0;i<NoC;i++)
	mu*=2;
rec=(__int8 *)malloc(mu);
rec[0]=1;
head[1]=0;
for(i=2;i<NoC;i++)
{
	head[i]=last;
	for(ptr=head[i-1];ptr<head[i];ptr+=(i-1))
	{
		rec[last]=i;
		for(j=0;j<i-1;j++)
			rec[last+j+1]=rec[ptr+j];
		last+=i;
	}
	for(j=i-2;3*j>i-2;j--)
	{
		for(ptr=head[j];ptr<head[j+1];ptr+=j)
		{
			if(2*j>i-2)
			{
				for(ptr2=head[i-j-1];ptr2<head[i-j];ptr2+=(i-j-1))
				{
					rec[last]=i;
					for(k=0;k<j;k++)
						rec[last+k+1]=rec[ptr+k];
					last+=j+1;
					for(k=0;k<i-j-1;k++)
						rec[last+k]=rec[ptr2+k];
					last+=(i-j-1);
				}
			}
			k=i-j-2;
			if(k>j)
				k=j;
			for(;2*k>i-j-2;k--)
			{
				for(ptr2=head[k];ptr2<head[k+1];ptr2+=k)
				{
					for(ptr3=head[i-j-k-1];ptr3<head[i-j-k];ptr3+=(i-j-k-1))
					{
						rec[last]=i;
						for(l=0;l<j;l++)
							rec[last+l+1]=rec[ptr+l];
						last+=j+1;
						for(l=0;l<k;l++)
							rec[last+l]=rec[ptr2+l];
```

```
                                        last+=k;
                                        for(l=0;l<i-j-k-1;l++)
                                                rec[last+l]=rec[ptr3+l];
                                        last+=(i-j-k-1);
                                }
                        }
                }
        }
        head[NoC]=last;
        for(ptr=head[NoC-1];ptr<head[NoC];ptr+=(NoC-1))
        {
                rec[last]=NoC;
                for(j=0;j<NoC-1;j++)
                        rec[last+j+1]=rec[ptr+j];
                check(last+NoC);
        }
        found2+=found1;
        found3+=found2;
        printf("%d isomers of C%dH%d found.\n",found,NoC,2*NoC+2);
        printf("%d isomers of C%dH%dSi found.\n",found1,NoC-1,2*NoC+2);
        printf("%d isomers of C%dH%dN found.\n",found2,NoC-1,2*NoC+1);
        printf("%d isomers of C%dH%dCl found.\n",found2,NoC,2*NoC+1);
        printf("%d isomers of C%dH%dO found.\n",found3,NoC-1,2*NoC);
        printf("Memory used:%d bytes.\n",last);
        system("pause");
}
```

附件三

馬德隆常數的程式設計

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int level=-1;
double method1()
{
	int i,j,k,n;
	double C[5];
	C[0]=0;
	for(i=1;i<=level;i++)
	{
		for(j=0;j<=i;j++)
		{
			for(k=0;k<=j;k++)
			{
				if(k>0)
				{
					if(j>k)
					{
						if(i>j)

	C[0]+=48.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
						else

	C[0]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
					}
					else
					{
						if(i>j)

	C[0]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
						else

	C[0]+=8.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
					}
				}
				else
				{
					if(j>k)
					{
						if(i>j)

	C[0]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
						else

	C[0]+=12.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
					}
					else
					{
	C[0]+=6.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
					}
				}
			}
		}
	}
	for(n=1;n<5;n++)
	{
		C[n]=C[n-1];
```

```
        for(j=0;j<=i;j++)
        {
                for(k=0;k<=j;k++)
                {
                        if(k>0)
                        {
                                if(j>k)
                                {
                                        if(i>j)
C[n]+=48.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                        else
C[n]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                                else
                                {
                                        if(i>j)
C[n]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                        else
C[n]+=8.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                        }
                        else
                        {
                                if(j>k)
                                {
                                        if(i>j)
C[n]+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                        else
C[n]+=12.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                                else
                                {
C[n]+=6.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                        }
                }
        }
        i++;
}
        return (C[0]+4*C[1]+6*C[2]+4*C[3]+C[4])/16;
}
double method2()
{
        int i,j,k,l,n;
        double C[5];
        C[0]=0;
        for(i=1;i<level;i++)
        {
                for(j=0;3*j<=i;j++)
                {
                        for(k=j;j+2*k<=i;k++)
                        {
                                l=i-j-k;
```

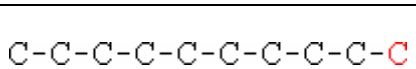
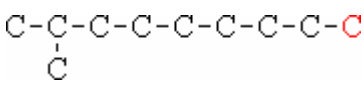
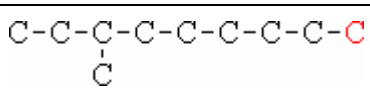
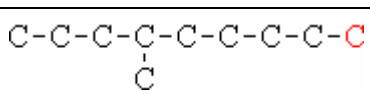
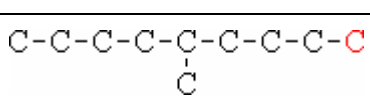
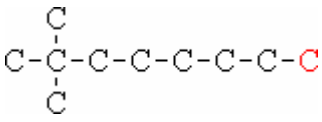
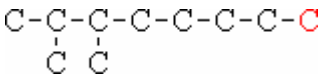
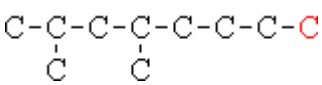
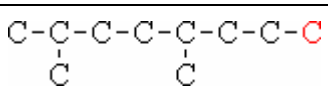
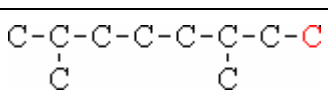
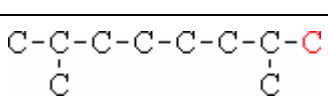
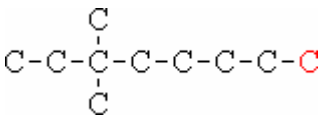
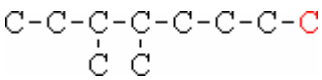
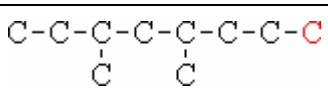
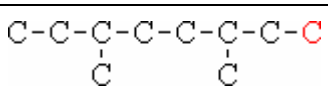
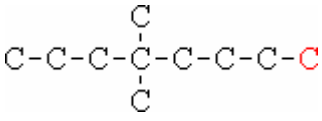
```
                    if(j>0)
                    {
                        if(k>j)
                        {
                            if(l>k)

C[0]+=48.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                            else

C[0]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                        }
                        else
                        {
                            if(l>k)

C[0]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                            else
C[0]+=8.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                        }
                    }
                    else
                    {
                        if(k>j)
                        {
                            if(l>k)

C[0]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                            else

C[0]+=12.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                        }
                        else
                        {
C[0]+=6.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                        }
                    }
                }
            }
}
for(n=1;n<5;n++)
{
        C[n]=C[n-1];
        for(j=0;3*j<=i;j++)
        {
            for(k=j;j+2*k<=i;k++)
            {
                l=i-j-k;
                if(j>0)
                {
                    if(k>j)
                    {
                        if(l>k)

C[n]+=48.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                            else

C[n]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                        }
                        else
```

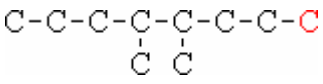
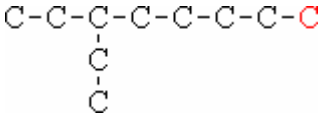
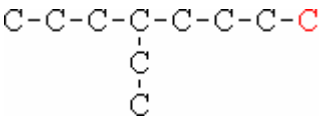
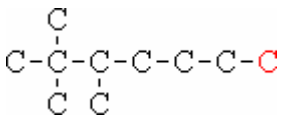
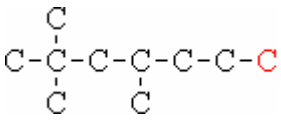
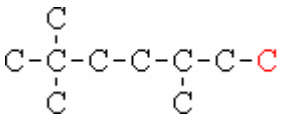
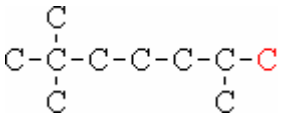
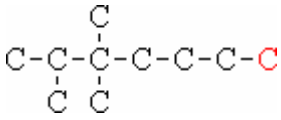
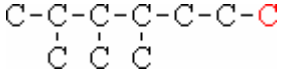
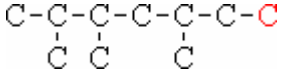
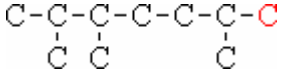
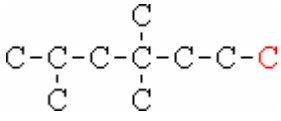
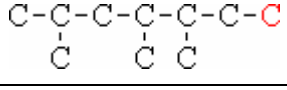
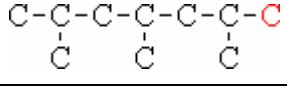
```
                                {
                                        if(l>k)

        C[n]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                                        else
        C[n]+=8.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                                }
                        }
                        else
                        {
                                if(k>j)
                                {
                                        if(l>k)

        C[n]+=24.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                                        else

        C[n]+=12.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                                }
                                else
                                {
        C[n]+=6.0*(double)((i%2)*2-1)/sqrt((double)(j*j+k*k+l*l));
                                }
                        }
                }
        }
        i++;
    }
    return (C[0]+4*C[1]+6*C[2]+4*C[3]+C[4])/16;
}
double mul[30000];
void runmul()
{
    int n=level*level*100;
    for(int i=1;i<20*level;i++)
    {
        if(i>10*level)
            n-=i-10*level;
        else
            n-=i;
        mul[i]=((double)n)/((double)level*level*100);
    }
}
double method3()
{
    int i,j,k;
    double C=0,m;
    runmul();
    for(i=1;i*i<level*level+10*level;i++)
    {
        for(j=0;j<=i && i*i+j*j<level*level+10*level;j++)
        {
            for(k=0;k<=j && i*i+j*j+k*k<=level*level-10*level;k++)
            {
                if(k>0)
                {
                    if(j>k)
                    {
                        if(i>j)
```

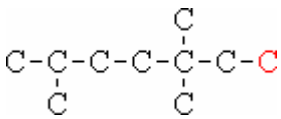
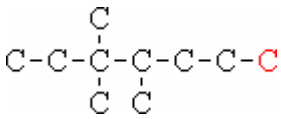
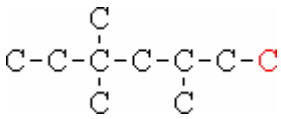
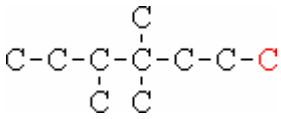
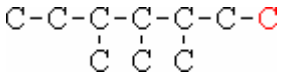
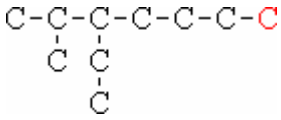
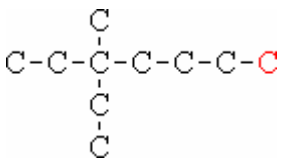
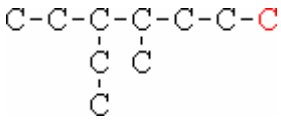
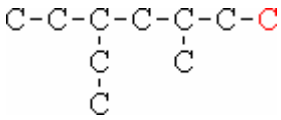
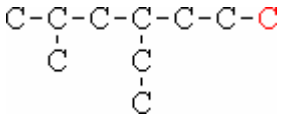
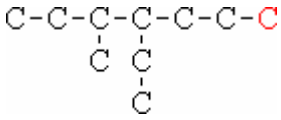
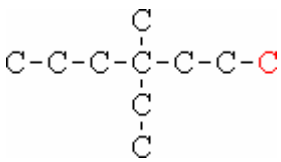
```
C+=48.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                else

C+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                        else
                        {
                                if(i>j)

C+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                else

C+=8.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                }
                else
                {
                        if(j>k)
                        {
                                if(i>j)

C+=24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                else

C+=12.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                        else
                        {
C+=6.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                }
        }
        for(;k<=j && i*i+j*j+k*k<level*level+10*level;k++)
        {
                m=mul[i*i+j*j+k*k-level*level+10*level];
                if(k>0)
                {
                        if(j>k)
                        {
                                if(i>j)

C+=m*48.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                else

C+=m*24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                        else
                        {
                                if(i>j)

C+=m*24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                else

C+=m*8.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                        }
                }
                else
                {
```

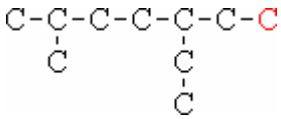
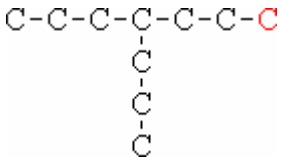
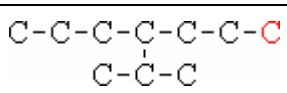
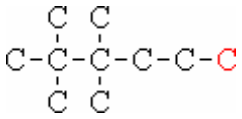
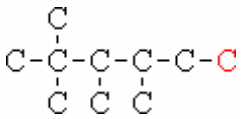
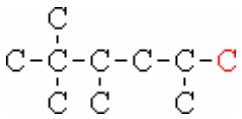
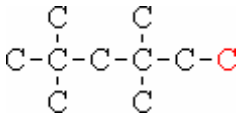
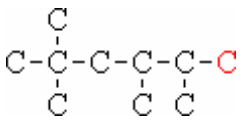
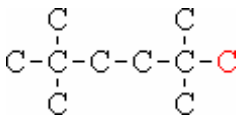
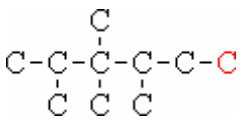
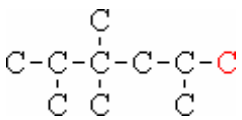
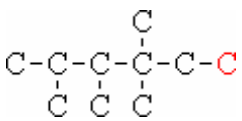
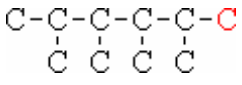
```
                                if(j>k)
                                {
                                        if(i>j)

        C+=m*24.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                        else

        C+=m*12.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                                else
                                {
        C+=m*6.0*(double)(((i+j+k)%2)*2-1)/sqrt((double)(i*i+j*j+k*k));
                                }
                        }
                }
        }
    }
    return C;
}
void main()
{
    while((level-1200)*level>0)
    {
        printf("search level=");
        scanf("%d",&level);
    }
    printf("%.12lf\n",method1());
    printf("%.12lf\n",method2());
    printf("%.12lf\n",method3());
    system("pause");
}
```

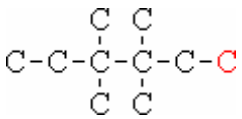
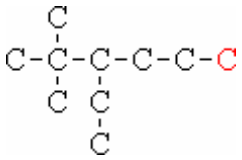
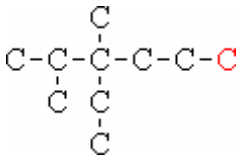
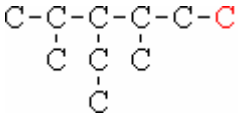
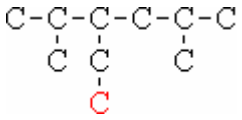
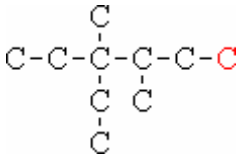
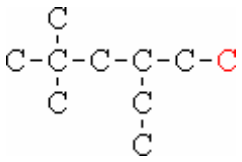
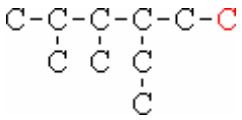
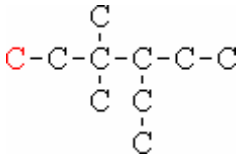
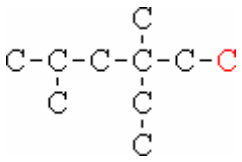
附件四

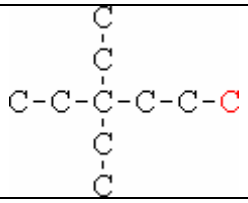
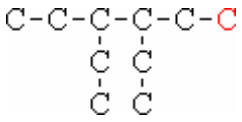
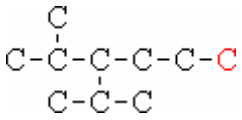
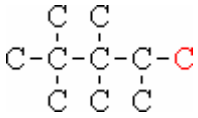
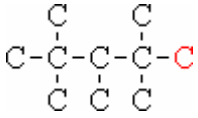
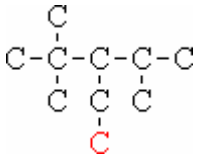
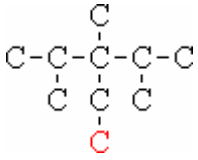
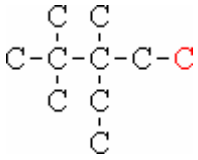
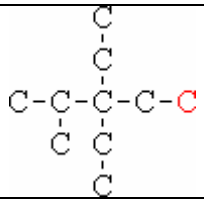
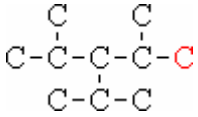
編號	結構式 (只顯示 C 與其間之鍵結)	IUPAC 系統命名	程式編碼	光學異構物
1		Decane 癸烷	10987654321	
2		2-Methylnonane 2-甲基壬烷	10987654311	
3		3-Methylnonane 3-甲基壬烷	10987654211	R/S
4		4-Methylnonane 4-甲基壬烷	10987653211	R/S
5		5-Methylnonane 5-甲基壬烷	10987643211	
6		2,2-Dimethyloctane 2,2-二甲基辛烷	10987654111	
7		2,3-Dimethyloctane 2,3-二甲基辛烷	10987653111	R/S
8		2,4-Dimethyloctane 2,4-二甲基辛烷	10987643111	R/S
9		2,5-Dimethyloctane 2,5-二甲基辛烷	10987543111	R/S
10		2,6-Dimethyloctane 2,6-二甲基辛烷	10986543111	R/S
11		2,7-Dimethyloctane 2,7-二甲基辛烷	10976543111	
12		3,3-Dimethyloctane 3,3-二甲基辛烷	10987652111	
13		3,4-Dimethyloctane 3,4-二甲基辛烷	10987642111	RR/SS//RS/SR
14		3,5-Dimethyloctane 3,5-二甲基辛烷	10987542111	RR/SS//RS/SR
15		3,6-Dimethyloctane 3,6-二甲基辛烷	10986542111	RR/SS//meso
16		4,4-Dimethyloctane 4,4-二甲基辛烷	10987632111	

17		4,5-Dimethyloctane 4,5-二甲基辛烷	10987532111	RR/SS//meso
18		3-Ethyloctane 3-乙基辛烷	10987652121	
19		4-Ethyloctane 4-乙基辛烷	10987632121	R/S
20		2,2,3-Trimethylheptane 2,2,3-三甲基庚烷	10987641111	R/S
21		2,2,4-Trimethylheptane 2,2,4-三甲基庚烷	10987541111	R/S
22		2,2,5-Trimethylheptane 2,2,5-三甲基庚烷	10986541111	R/S
23		2,2,6-Trimethylheptane 2,2,6-三甲基庚烷	10976541111	
24		2,3,3-Trimethylheptane 2,3,3-三甲基庚烷	10987631111	
25		2,3,4-Trimethylheptane 2,3,4-三甲基庚烷	10987531111	RR/SS//RS/SR
26		2,3,5-Trimethylheptane 2,3,5-三甲基庚烷	10986531111	RR/SS//RS/SR
27		2,3,6-Trimethylheptane 2,3,6-三甲基庚烷	10976531111	R/S
28		2,4,4-Trimethylheptane 2,4,4-三甲基庚烷	10987431111	
29		2,4,5-Trimethylheptane 2,4,5-三甲基庚烷	10986431111	RR/SS//RS/SR
30		2,4,6-Trimethylheptane 2,4,6-三甲基庚烷	10976431111	

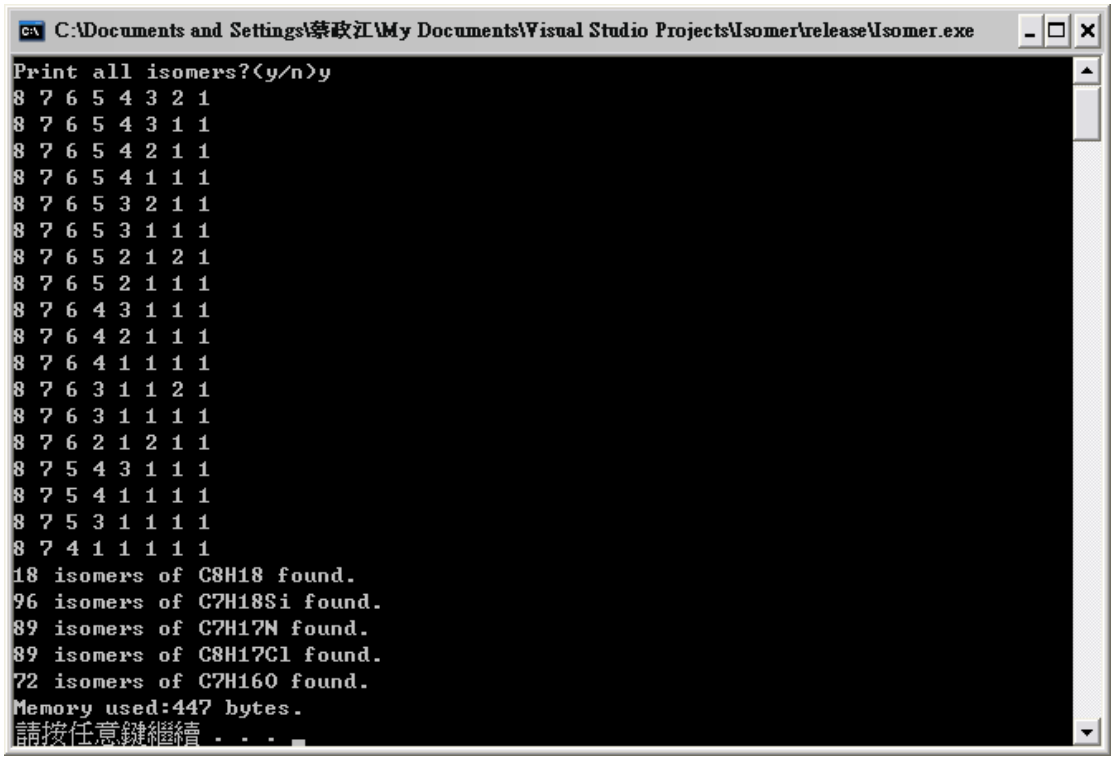
31		2,5,5-Trimethylheptane 2,5,5-三甲基庚烷	10985431111	
32		3,3,4-Trimethylheptane 3,3,4-三甲基庚烷	10987521111	R/S
33		3,3,5-Trimethylheptane 3,3,5-三甲基庚烷	10986521111	R/S
34		3,4,4-Trimethylheptane 3,4,4-三甲基庚烷	10987421111	R/S
35		3,4,5-Trimethylheptane 3,4,5-三甲基庚烷	10986421111	RR/SS//meso
36		3-Ethyl-2-methylheptane 3-乙基-2-甲基庚烷	10987631121	
37		3-Ethyl-3-methylheptane 3-乙基-3-甲基庚烷	10987621211	
38		3-Ethyl-4-methylheptane 3-乙基-4-甲基庚烷	10987521211	R/S
39		3-Ethyl-5-methylheptane 3-乙基-5-甲基庚烷	10986521211	R/S
40		4-Ethyl-2-methylheptane 4-乙基-2-甲基庚烷	10987431121	R/S
41		4-Ethyl-3-methylheptane 4-乙基-3-甲基庚烷	10987421121	RR/SS//RS/SR
42		4-Ethyl-4-methylheptane 4-乙基-4-甲基庚烷	10987321211	

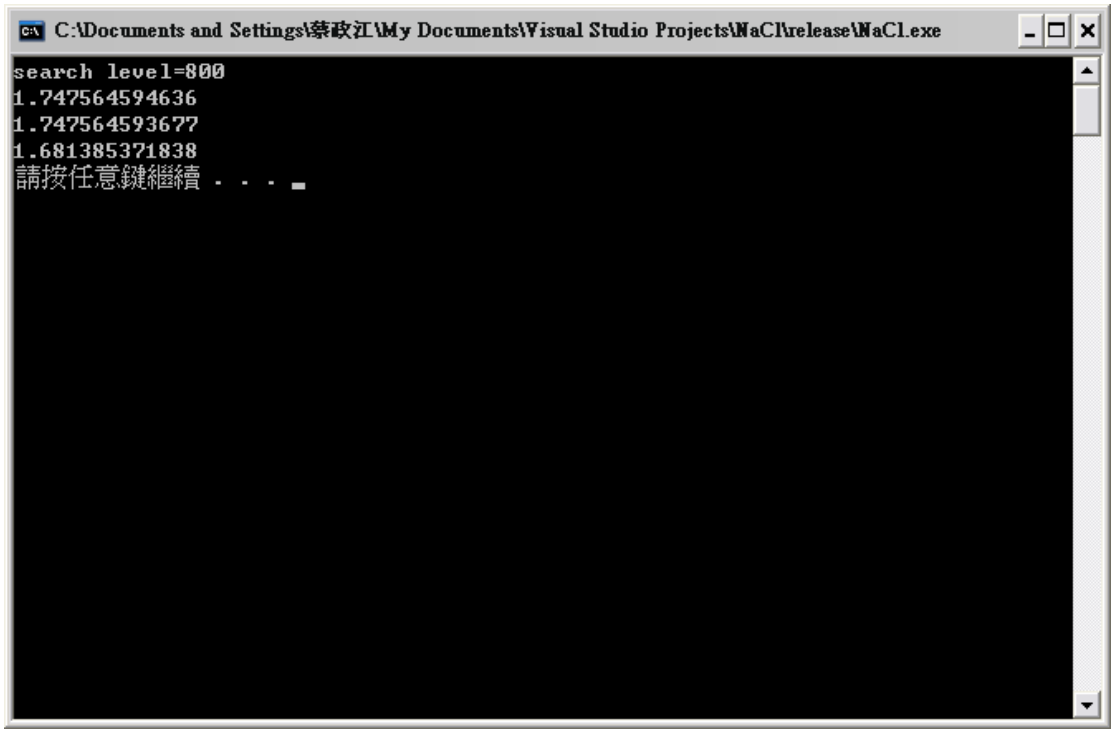
43		5-Ethyl-2-methylheptane 5-乙基-2-甲基庚烷	10985431121	
44		4-Propylheptane 4-正丙基庚烷	10987321321	
45		4-Isopropylheptane 4-异丙基庚烷	10987321311	
46		2,2,3,3-Tetramethylhexane 2,2,3,3-四甲基己烷	10987411111	
47		2,2,3,4-Tetramethylhexane 2,2,3,4-四甲基己烷	10986411111	RR/SS//RS/SR
48		2,2,3,5-Tetramethylhexane 2,2,3,5-四甲基己烷	10976411111	R/S
49		2,2,4,4-Tetramethylhexane 2,2,4,4-四甲基己烷	10985411111	
50		2,2,4,5-Tetramethylhexane 2,2,4,5-四甲基己烷	10975411111	R/S
51		2,2,5,5-Tetramethylhexane 2,2,5,5-四甲基己烷	10965411111	
52		2,3,3,4-Tetramethylhexane 2,3,3,4-四甲基己烷	10986311111	R/S
53		2,3,3,5-Tetramethylhexane 2,3,3,5-四甲基己烷	10976311111	
54		2,3,4,4-Tetramethylhexane 2,3,4,4-四甲基己烷	10985311111	R/S
55		2,3,4,5-Tetramethylhexane 2,3,4,5-四甲基己烷	10975311111	RR/SS//meso

56		3,3,4,4-Tetramethylhexane 3,3,4,4-四甲基己烷	10985211111	
57		3-Ethyl-2,2-dimethylhexane 3-乙基-2,2-二甲基己烷	10987411121	R/S
58		3-Ethyl-2,3-dimethylhexane 3-乙基-2,3-二甲基己烷	10987311211	R/S
59		3-Ethyl-2,4-dimethylhexane 3-乙基-2,4-二甲基己烷	10986311211	RR/SS//RS/SR
60		3-Ethyl-2,5-dimethylhexane 3-乙基-2,5-二甲基己烷	10984311311	R/S
61		3-Ethyl-3,4-dimethylhexane 3-乙基-3,4-二甲基己烷	10986212111	R/S
62		4-Ethyl-2,2-dimethylhexane 4-乙基-2,2-二甲基己烷	10985411121	
63		4-Ethyl-2,3-dimethylhexane 4-乙基-2,3-二甲基己烷	10985311121	R/S
64		4-Ethyl-3,3-dimethylhexane 4-乙基-3,3-二甲基己烷	10985212111	
65		4-Ethyl-2,4-dimethylhexane 4-乙基-2,4-二甲基己烷	10984311211	

66		3,3-Diethylhexane 3,3-二乙基己烷	10987212121	
67		3,4-Diethylhexane 3,4-二乙基己烷	10985212121	
68		3-Isopropyl-2-methylhexane 3-異丙基-2-甲基己烷	10987311311	
69		2,2,3,3,4-Pentamethylpentane 2,2,3,3,4-五甲基戊烷	10974111111	
70		2,2,3,4,4-Pentamethylpentane 2,2,3,4,4-五甲基戊烷	10964111111	
71		3-Ethyl-2,2,4-trimethylpentane 3-乙基-2,2,4-三甲基戊烷	10984111311	R/S
72		3-Ethyl-2,3,4-trimethylpentane 3-乙基-2,3,4-三甲基戊烷	10983113111	
73		3-Ethyl-2,2,3-trimethylpentane 3-乙基-2,2,3-三甲基戊烷	10984111211	
74		3,3-Diethyl-2-methylpentane 3,3-二乙基-2-甲基戊烷	10983112121	
75		3-Isopropyl-2,4-dimethylpentane 3-異丙基-2,4-二甲基戊烷	10973113111	

附件五


C:\Documents and Settings\蔡政江\My Documents\Visual Studio Projects\Isomer\release\Isomer.exe
Print all isomers?(y/n)y
8 7 6 5 4 3 2 1
8 7 6 5 4 3 1 1
8 7 6 5 4 2 1 1
8 7 6 5 4 1 1 1
8 7 6 5 3 2 1 1
8 7 6 5 3 1 1 1
8 7 6 5 2 1 2 1
8 7 6 5 2 1 1 1
8 7 6 4 3 1 1 1
8 7 6 4 2 1 1 1
8 7 6 4 1 1 1 1
8 7 6 3 1 1 2 1
8 7 6 3 1 1 1 1
8 7 6 2 1 2 1 1
8 7 5 4 3 1 1 1
8 7 5 4 1 1 1 1
8 7 5 3 1 1 1 1
8 7 4 1 1 1 1 1
18 isomers of C8H18 found.
96 isomers of C7H18Si found.
89 isomers of C7H17N found.
89 isomers of C8H17Cl found.
72 isomers of C7H16O found.
Memory used:447 bytes.
請按任意鍵繼續 . . .


C:\Documents and Settings\蔡政江\My Documents\Visual Studio Projects\NaCl\release\NaCl.exe
search level=800
1.747564594636
1.747564593677
1.681385371838
請按任意鍵繼續 . . .

評語

本實驗利用電腦輔助，自行撰寫程式，由碳的數目即可找出其結構異構物的數目，相當有創意，故推薦爲第三名。