

鏈結串列自編教材（一）

本教材（一）目標問題：每次以亂數產生一 $[0,1000]$ 之整數值，若該值 >100 ，則以同方式繼續產生下一亂數值，若該值 ≤ 100 ，則停止此產生值之程序。所有產生之值必須全部記錄，並可重複列印所有產生的值。

試解：定義一夠大之整數陣列，儲印所有產生之值。

疑問：整數陣列必須宣告多少才夠大？以機率考量，預設固定大小之陣列一定不足！（設過大整數陣列，通常可執行但浪費系統資源）

正解：以 linked list 逐一記錄產生之值

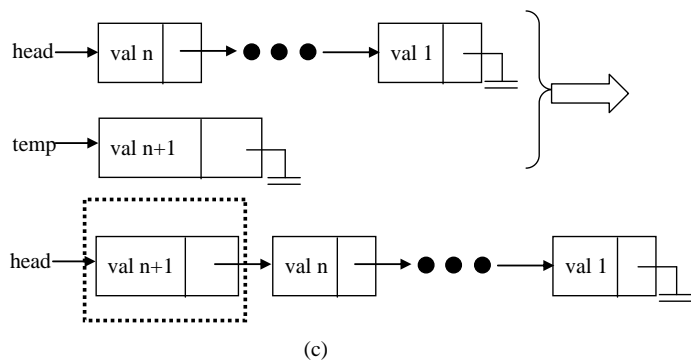
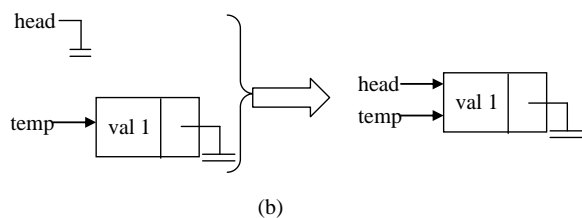
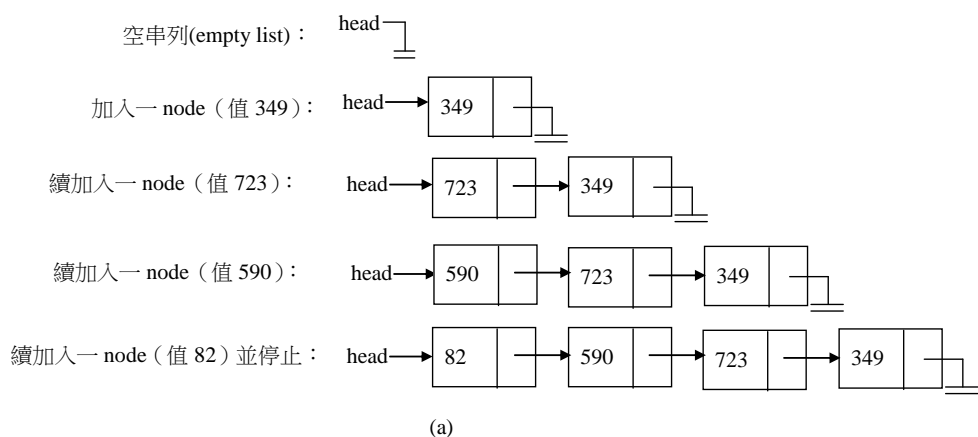


圖 1

● C 語言如何用亂數產生 1 個特定範圍（如[0,1000]）的整數？(s3-1.c)

```
-----  
#include <stdlib.h> //包含定義 srand()和 rand()  
#include <time.h> //包含定義 time()  
  
srand(time(NULL)); //設亂數種子（使用現在的系統時間，首次產生亂數前用）  
printf("%d\n",rand()%1001);  
-----
```

● C 語言如何用亂數產生 n 個特定範圍（如[10,100]）的整數？(s3-2.c)


```
-----  
//定義同 s3-1.c  
  
srand(time(NULL)); //設亂數種子（使用現在的系統時間，首次產生亂數前用）  
scanf("%d",&n);  
for (i=0;i<n;i++)  
    printf("%d\n",rand()%91+10);  
-----
```

● C 語言如何每次以亂數產生一[0,1000]之整數值，若該值>100，則以同方式繼續產生下一亂數值，若該值<=100，則停止此產生值之程序。(s3-3.c)

```
-----  
//定義同 s3-1.c  
int i;  
srand(time(NULL)); //設亂數種子（使用現在的系統時間，首次產生亂數前用）  
do{  
    i=rand()%1001;  
    printf("%d\n",i);  
} while (i>100);  
-----
```

- 宣告結構 struct ANODE（其中包含一個整數變數 val，及一個指向 struct ANODE 的指標變數 next）

```
-----  
struct ANODE{  
    int val;  
    struct ANODE *next;  
};  
struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列
```

head 

temp

圖 2

- 指標 temp 指向一新產生的 struct ANODE 結構，並在 val 中存放值 349，及將 next 指標內容設為 NULL

```
temp=(struct ANODE *)malloc(sizeof(struct ANODE));  
temp->val=349;  
temp->next=NULL;
```

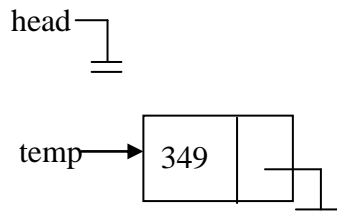


圖 3

- 當指標 head 原指向 NULL(即 head 指向空串列)，將指標 head 指向指標 temp 所指，即完成在空串列中加入第 1 個 node

```
head=temp;
```

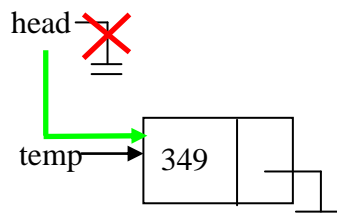


圖 4

- 在空串列中加入第 1 個 node（值為 349）的範例(s3-4.c)

```
-----  
struct ANODE{  
    int val;  
    struct ANODE *next;  
};  
struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列  
  
//建立新 node，並由 temp 所指，設定 node 的變數內容  
temp=(struct ANODE *)malloc(sizeof(struct ANODE));  
temp->val=349;  
temp->next=NULL;//NULL 要大寫  
  
//指標 head 指向指標 temp，完成在空串列加入第 1 個 node  
head=temp;  
  
//印出串列第 1 個 node 的 val 值  
printf(“%d %d\n”,head->val,temp->val);  
-----
```

- 空串列加入第 1 個 node（值為亂數產生[0,1000]的整數）的範例(s3-5.c)

```
-----  
struct ANODE{  
    int val;  
    struct ANODE *next;  
};  
struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列  
  
srand(time(NULL));//設亂數種子（使用現在的系統時間，首次產生亂數前用）  
//建立新 node，並由 temp 所指，設定 node 的變數內容  
temp=(struct ANODE *)malloc(sizeof(struct ANODE));  
temp->val= rand()%1001;  
temp->next=NULL;  
//指標 head 指向指標 temp，完成在空串列加入第 1 個 node  
head=temp;  
//印出串列第 1 個 node 的 val 值  
printf(“%d %d\n”,head->val,temp->val);  
-----
```

- 在非空串列開頭加入一個新 node（值為亂數產生[0,1000]的整數）(s3-6.c)

//指令有順序性

temp->next=head;//(1)

head=temp;//(2)

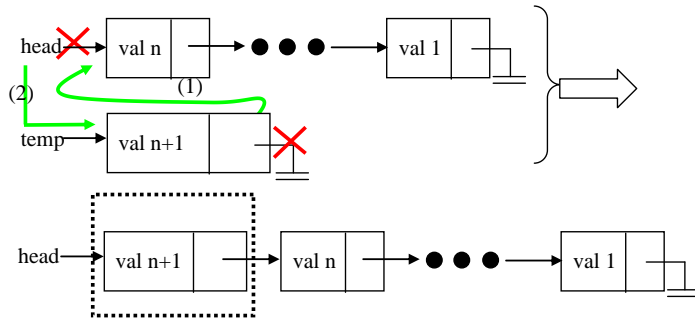


圖 5

- 在非空串列開頭加入 1 個新 node（值為亂數產生[0,1000]的整數）的範例程式(s3-6.c)

```
-----  
struct ANODE{
```

```
    int val;
```

```
    struct ANODE *next;
```

```
};
```

```
struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列
```

```
srand(time(NULL));//設亂數種子(使用現在的系統時間，首次產生亂數前用)
```

```
//先產生一個非空串列（只有一個 node）
```

```
temp=(struct ANODE *)malloc(sizeof(struct ANODE));
```

```
temp->val= rand()% 1001;
```

```
temp->next=NULL;
```

```
head=temp;
```

```
printf(“%d %d\n”,head->val,temp->val);
```

```
//產生一個新的 node
```

```
temp=(struct ANODE *)malloc(sizeof(struct ANODE));
```

```
temp->val= rand()% 1001;
```

```
temp->next=NULL;
```

```
//將新產生的 node 加入串列的開頭
temp->next=head;//(1)
head=temp;//(2)

//印出串列中 2 個 node 的值
printf(“%d %d\n”,head->val,head->next->val);
-----
```

- 適當的印出非空串列的前 2 個 node 之值
//ptr 指向串列的第 2 個 node
ptr=head->next;//要宣告 ptr 為指向 struct ANODE 的指標
printf(“%d %d\n”,head->val,ptr->val);

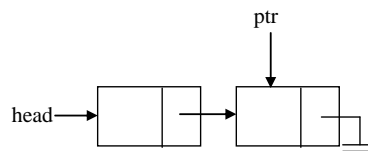


圖 6

- 空串列陸續由串列開頭加入 10 個 node（值為亂數產生[0,1000]的整數）
(s3-7.c)

```
-----  
struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列  
srand(time(NULL));//設亂數種子（使用現在的系統時間，首次產生亂數前用）
```

```
//產生一個新的 node
```

```
temp=(struct ANODE *)malloc(sizeof(struct ANODE));
```

```
temp->val= rand()% 1001;
```

```
temp->next=NULL;
```

```
//加入第 1 個 node
```

```
head=temp;
```

```
//加入第 2~10 個 node
```

```
for (i=0;i<9;i++){
```

```
    //產生一個新的 node
```

```
    temp=(struct ANODE *)malloc(sizeof(struct ANODE));
```

```
    temp->val= rand()% 1001;
```

```
    temp->next=NULL;
```

```
    //將新產生的 node 加入串列的開頭
```

```
    temp->next=head;//(1)
```

```
    head=temp;//(2)
```

```
}
```

```
//列印第 10,9,...,1 個 node 的值
```

```
ptr=head;
```

```
while (ptr!=NULL){
```

```
    printf("%d\n",ptr->val);
```

```
    ptr=ptr->next;
```

```
}
```

● 上一範例整理為函數呼叫(s3-8.c)

struct ANODE *head=NULL,*temp;//串列由 head 所指，預設為空串列
srand(time(NULL));//設亂數種子(使用現在的系統時間，首次產生亂數前用)

//generateanode 函數每次產生一個新 node，並回傳新 node 的起始位址

```
struct ANODE * generateanode(){  
    struct ANODE * ptr;  
    ptr=(struct ANODE *)malloc(sizeof(struct ANODE));  
    ptr->val= rand()%1001;  
    ptr->next=NULL;  
    return ptr;//此為函數的回傳值，指向配置的記憶體起始位址  
}
```

//printlist 函數依序列印串列中每個 node 的值

```
void printlist(){  
    struct ANODE * ptr;  
    ptr=head;  
    while (ptr!=NULL){  
        printf("%d\n",ptr->val);  
        ptr=ptr->next;  
    }  
}
```

```
main(){  
    //呼叫函數 generateanode()產生一個 node  
    temp=generateanode();
```

```
    //加入第 1 個 node  
    head=temp;
```

```
    //加入第 2~10 個 node  
    for (i=0;i<9;i++){
```

```
        //產生一個新的 node  
        temp= generateanode();
```

```
        //將新產生的 node 加入串列的開頭
```



```
        temp->next=head;//(1)
        head=temp;//(2)

    }

    //列印第 10,9,...,1 個 node 的值
    printlist();
}
-----
```

- 將上一範例重新整理，使用同一 for 迴圈可產生所有 node 及加入串列(s3-9.c)

```
-----  
main(){  
  
    //加入第 1~10 個 node  
    for (i=0;i<10;i++){  
        //產生一個新的 node  
        temp= generateanode();  
  
        if (head==NULL)//空串列  
            head=temp;  
        else{//將新產生的 node 加入非空串列的開頭  
            temp->next=head;//(1)  
            head=temp;//(2)  
        }  
    }  
  
    //列印第 10,9,...,1 個 node 的值  
    printlist();  
}  
-----
```

- 整理上一範例，將 for 迴圈中的 if 條件移除，即可逐一產生所有的新 node 及加入串列(s3-10.c)

```
-----  
main(){  
  
    //加入第 1~10 個 node  
    for (i=0;i<10;i++){  
        //產生一個新的 node  
        temp= generateanode();  
  
        //將新產生的 node 加入串列的開頭  
        temp->next=head;//(1)  
        head=temp;//(2)  
    }  
  
    //列印第 10,9,...,1 個 node 的值  
    printlist();  
}  
-----
```

- 每次以亂數產生一[0,1000]之整數值，若該值>100，則以同方式繼續產生下一亂數值，若該值≤100，則停止此產生值之程序。所有產生之值必須全部記錄，並可重複列印所有產生的值。(s3-11.c)

```
-----  
//若加入新 node 之值>100，則繼續產生下一新 node  
do{  
    //產生一個新的 node  
    temp= generateanode();  
  
    //將新產生的 node 加入串列的開頭  
    temp->next=head;//(1)  
    head=temp;//(2)  
} while (temp->val>100);  
-----
```