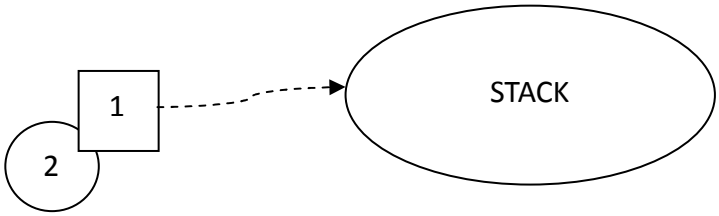
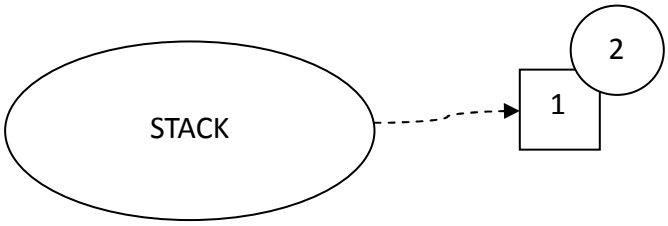
Ex. 1 以陣列實作"堆疊"來儲存資料.

堆疊原理: 後進先出



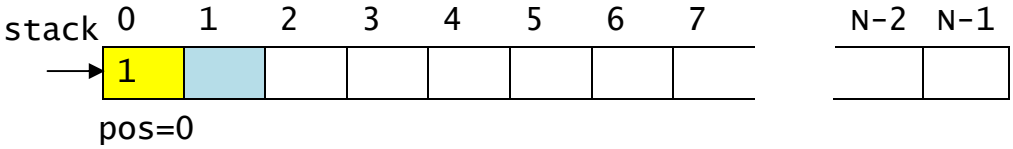
進入順序 1, 2



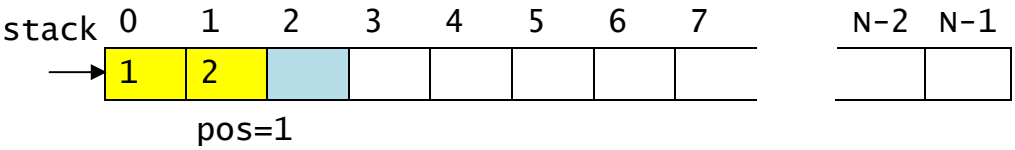
退出順序 2, 1

使用陣列實作

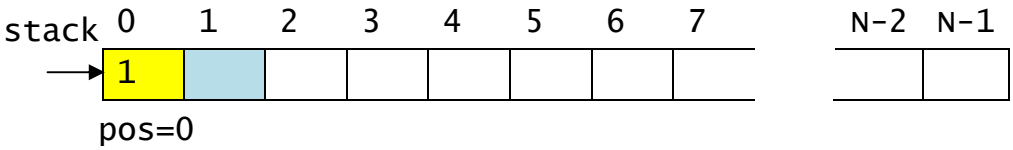
新增 (push) "1"



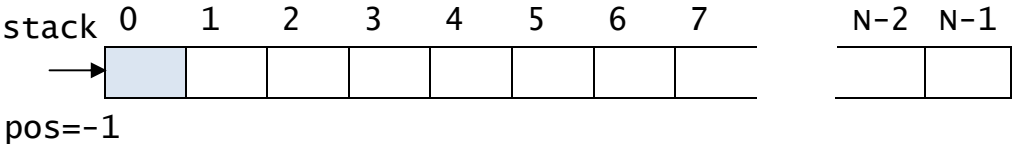
新增 (push) "2"



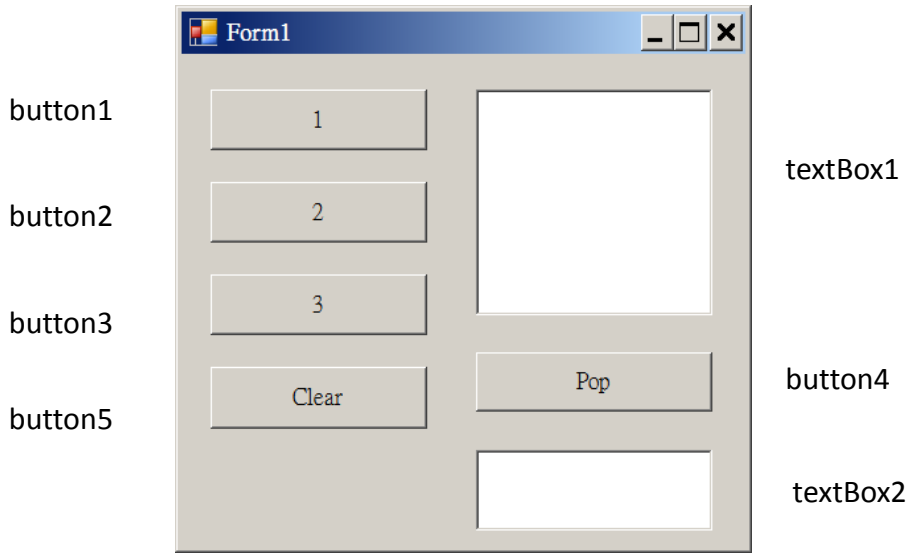
退出 (pop) "2"



退出 (pop) "1"



(程式設計)



(程式碼)

```
ref class STACK {
 private:
    int N;
    array<String^> ^name;
    int pos;
public:
    STACK(){
        N=5;
        pos=-1;
        name = gcnew array<String^>( N );
    }
    void push( String ^x ){   //新增資料
        if( pos >= N-1) {
            N = 2*N;
            array<String^> ^s
                  = gcnew array<String^>( N );
            int k;
            for(k=0; k< N/2; k++) s[k]=name[k];
            name = s;
        }
        ++pos;
        name[pos]= x;
    }
```

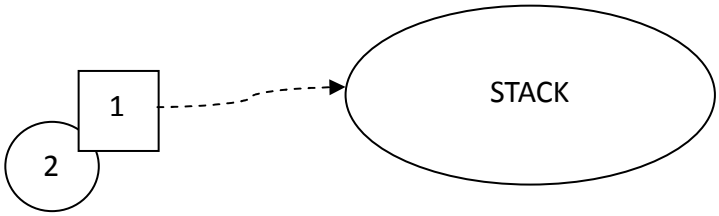
```
            String^ pop(){     //退出資料
                String ^x;
                if( pos>=0 ) {
                    x = name[pos];
                    --pos;
                } else x = "";
                return x;
            }
            String^ show(){
                String ^s="";
                int k;
                for(k=0; k<= pos; k++ )
                    s += "[" + name[k] + "]";
                return s;
            }
            void clear(){
                 pos=-1;
            }
        };
        STACK ^A;

private: System::Void Form1_Load(System::Object^  sender,
                                 System::EventArgs^  e) {
            A=gcnew STACK;
            textBox1->Text = A->show();
        }
private: System::Void button1_Click(System::Object^
                                sender, System::EventArgs^  e) {
            A->push( button1->Text );
            textBox1->Text = A->show();
        }
private: System::Void button2_Click(System::Object^
                                sender, System::EventArgs^  e) {
            A->push( button2->Text );
            textBox1->Text = A->show();
        }
private: System::Void button3_Click(System::Object^
                                sender, System::EventArgs^  e) {
```

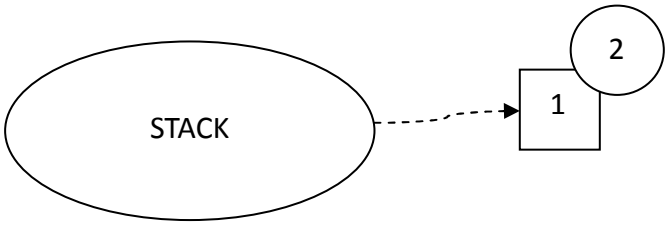
```
            A->push( button3->Text );
            textBox1->Text = A->show();
        }
private: System::Void button4_Click(System::Object^
                                sender, System::EventArgs^  e) {
            textBox2->Text = A->pop();
            textBox1->Text = A->show();
        }
private: System::Void button5_Click(System::Object^
                                sender, System::EventArgs^  e) {
            A->clear();
            textBox1->Text = A->show();
            textBox2->Text = "";
        }
```

Ex. 2 以串列實作"堆疊"來儲存資料.

堆疊原理: 後進先出



進入順序 1, 2



退出順序 2, 1

使用串列實作

空串列

(null)

新增 (push) "1"

"1"->(null)

新增 (push) "2"

"2"->"1"->(null)

退出 (pop) "2"

"1"->(null)

退出 (pop) "1"

(null)

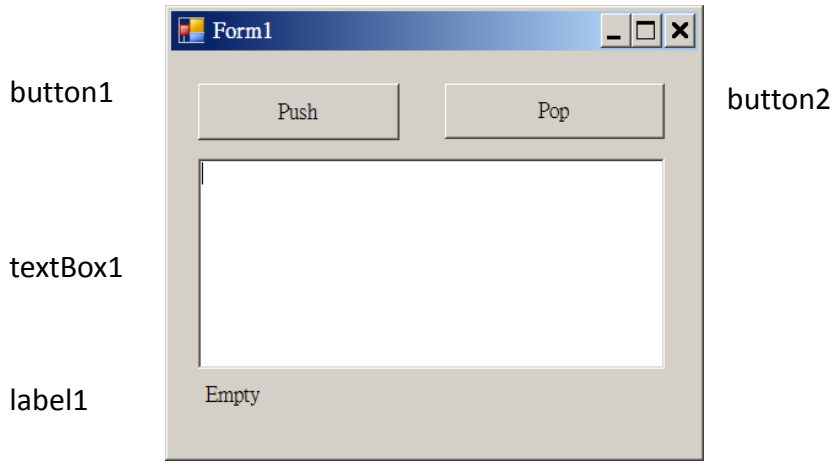
(程式碼)

```
ref class list {
 public:
    String ^name;
    list   ^next;
    list( String ^s ){
        name=s;
        next=nullptr;
    }
};
ref class STACK {
  private:
    list ^L;
  public:
    STACK() { L=nullptr; }
    void push( String ^s ){        //新增資料
        list ^x = gcnew list( s );
        x->next = L;
        L=x;
    }
    String^ pop(){                 //退出資料
        String ^s;
        if( L != nullptr ){
            s = L->name;
            L=L->next;
        } else s="";
        return s;
    }
    String^ show(){
        String ^s = "";
        list ^x=L;
        while( x !=nullptr ){
            s += x->name + "->";
            x=x->next;
        }
        s += "(null)";
        return s;
    }
```

```
            void clear(){ L=nullptr; }
        };

        STACK ^A;
private: System::Void Form1_Load(System::Object^  sender,
                        System::EventArgs^  e) {
            A=gcnew STACK;
            textBox1->Text = A->show();
        }
private: System::Void button1_Click(System::Object^
                        sender, System::EventArgs^  e) {
            A->push( button1->Text );
            textBox1->Text = A->show();
        }
private: System::Void button2_Click(System::Object^
                        sender, System::EventArgs^  e) {
            A->push( button2->Text );
            textBox1->Text = A->show();
        }
private: System::Void button3_Click(System::Object^
                        sender, System::EventArgs^  e) {
            A->push( button3->Text );
            textBox1->Text = A->show();
        }
private: System::Void button4_Click(System::Object^
                        sender, System::EventArgs^  e) {
            textBox2->Text = A->pop();
            textBox1->Text = A->show();
        }
private: System::Void button5_Click(System::Object^
                        sender, System::EventArgs^  e) {
            A->clear();
            textBox1->Text = A->show();
            textBox2->Text = "";
        }
```

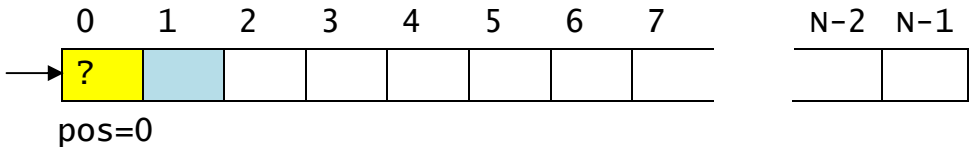
Ex. 3 以陣列實作有限長度堆疊，並列出相關資訊。



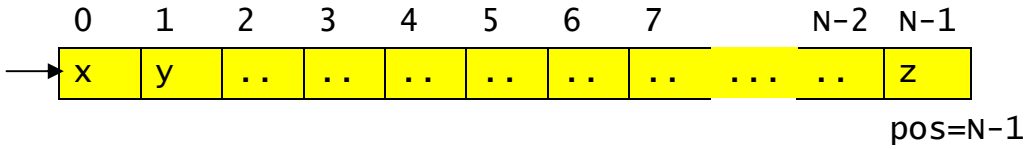
(空陣列)

pos = -1 表示空陣列

(新增資料)



(陣列全滿的情況)



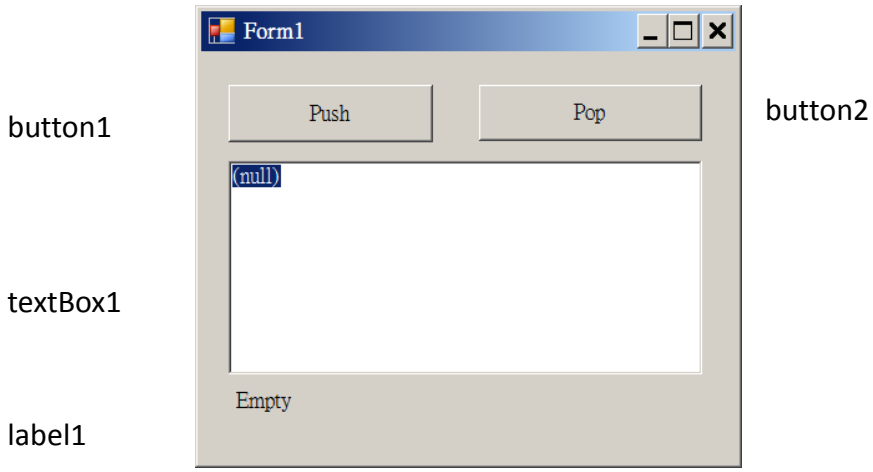
陣列已儲存元素總數為 pos+1

(程式碼)

```
ref class STACK {
 private:
    int N;
    array<String^> ^name;
    int pos;
public:
    STACK(){
        N=5;
        pos=-1;
        name = gcnew array<String^>( N );
    }
    void Push( String ^x ){   //新增資料
        if( pos < N-1) {
            ++pos;
            name[pos]= x;
        }
     }
    String^ Pop(){     //退出資料
        String ^x;
        if( pos>=0) {
            x = name[pos];
            --pos;
        } else x = "";
        return x;
    }
    void Clear(){
         pos=-1;
    }
    bool IsEmpty(){
        if( pos==-1 ) return true; else return false;
    }
    bool IsFull(){
        if( pos==N-1 ) return true; else return false;
    }
    int Total(){
        return pos+1;
    }
```

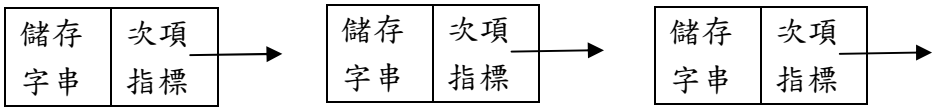
```
            String^ Show(){
                String ^s="";
                int k;
                for(k=0; k<= pos; k++ )
                    s += "[" + name[k] + "]";
                return s;
            }
        };
        STACK ^A;
        void report(){
            if( A->IsEmpty() ) label1->Text = "Empty";
            else if ( A->IsFull() ) label1->Text = "Full";
            else label1->Text = "Ready";
        }
private: System::Void Form1_Load(System::Object^  sender,
                    System::EventArgs^  e) {
            A=gcnew STACK;
            textBox1->Text = A->Show();
        }
private: System::Void button1_Click(System::Object^
                    sender, System::EventArgs^  e) {
            int x = A->Total() + 1;
            A->Push( x.ToString() );
            textBox1->Text = A->Show();
            report();
        }
private: System::Void button2_Click(System::Object^
                    sender, System::EventArgs^  e) {
            A->Pop();
            textBox1->Text = A->Show();
            report();
        }
```

Ex.4 同上，以串列實作堆疊，並列出相關資訊。



(資料結構)

串列節點 node




```
class list {
        String  ^name; //儲存字串
        list    ^next; //次項指標
} ^A;
```

開始時 A 指向無效指標(null)，代表空串列

(null)

加入 "1"

"1" -> (null)

加入 "2"

"2" -> "1" -> (null)

加入 "3"

"3" -> "2" -> "1" -> (null)

(程式碼)

```
ref class list {
 public:
    String ^name;
    list   ^next;
    list( String ^s ){
        name=s;
        next=nullptr;
    }
};
ref class STACK {
  private:
    list ^L;
    int n;
  public:
    STACK() { L=nullptr; n=0; }
    void Push( String ^s ){        //新增資料
        list ^x = gcnew list( s );
        x->next = L;
        L=x;
        ++n;
    }
    String^ Pop(){                 //退出資料
        String ^s;
        if( L != nullptr ){
            s = L->name;
            L=L->next;
            --n;
        } else s="";
        return s;
    }
    void Clear(){
         L=nullptr;    n=0;
    }
    bool IsEmpty(){
        if( L == nullptr ) return true;
        else return false;
    }
```

```
            int Total(){
                return n;
            }
            String^ Show(){
                String ^s = "";
                list ^x=L;
                while( x !=nullptr ){
                    s += x->name + "->";
                    x=x->next;
                }
                s += "(null)";
                return s;
            }
        };
        STACK ^A;
        void report(){
            if( A->IsEmpty() ) label1->Text = "Empty";
            else label1->Text = "Ready";
        }
private: System::Void Form1_Load(System::Object^  sender,
                    System::EventArgs^  e) {
            A=gcnew STACK;
            textBox1->Text = A->Show();
        }
private: System::Void button1_Click(System::Object^
                    sender, System::EventArgs^  e) {
            int x = A->Total() + 1;
            A->Push( x.ToString() );
            textBox1->Text = A->Show();
            report();
        }
private: System::Void button2_Click(System::Object^
                    sender, System::EventArgs^  e) {
            A->Pop();
            textBox1->Text = A->Show();
            report();
        }
```