

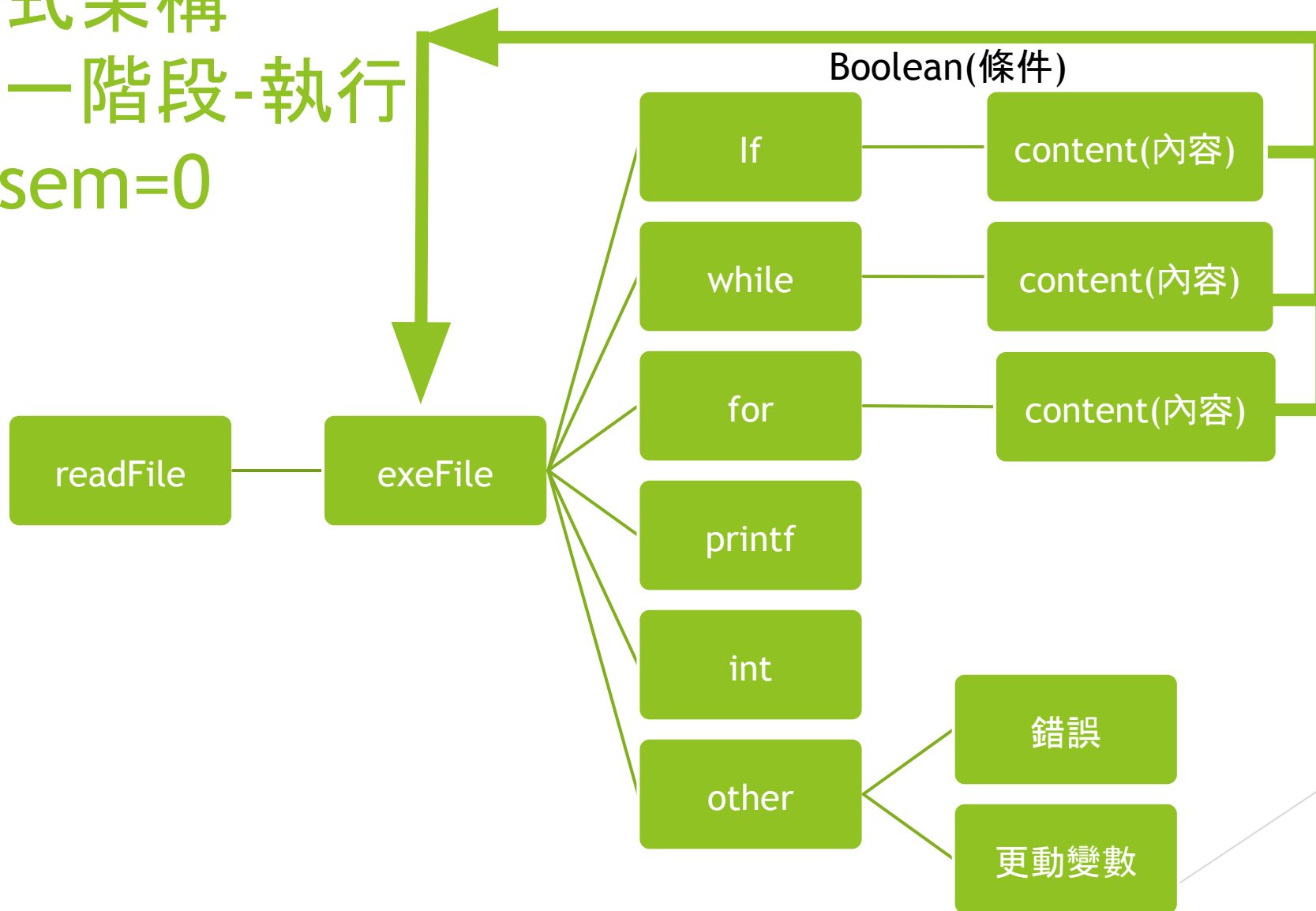
組合語言期末專題報告

組員：黃韻儒、賈慶郁、朱詠田、陳俊元

程式架構

第一階段-執行

assem=0



程式架構

- ▶ 呼叫readFile, 把整個txt的內文 放進exeFile中
- ▶ exeFile 有讀取指標 i
- ▶ exeFile 中會抓取關鍵字「int」、「if」、「while」、「for」、「print」

以上五個有各自的函式去執行與解讀

對if、while、for, 會呼叫函式把條件式和要執行的內文content抓取起來,
然後丟到各自對應的功能中

除了以上五個之外, 會丟到other函式

other函式會判斷第一個單字是否為變數, 若是, 則判定為更改函數的指令, 呼叫更新變數內容的函數updateValue

若否, 則為指令錯誤。

程式架構

- ▶ 對於if while for的讀取 &執行結構
- ▶ 呼叫if while for 時會丟入字串 條件式 Boolean 與 執行內容content
- ▶ 判斷條件成立後, 就會把 content 作為引數, 呼叫一開始的exefile

int 宣告

int_func(string str, bool assem)

- ▶ 宣告 int a; value 預設為 0

- ▶ 宣告 int b = 0;

↓
key



value_str



value=compute_int(value_str);

- ▶ if(is_variable_declare(key)){ 確認是否已經宣告過了

```
cout<<"int_redeclaration"<<endl;
```

```
is_error=true;
```

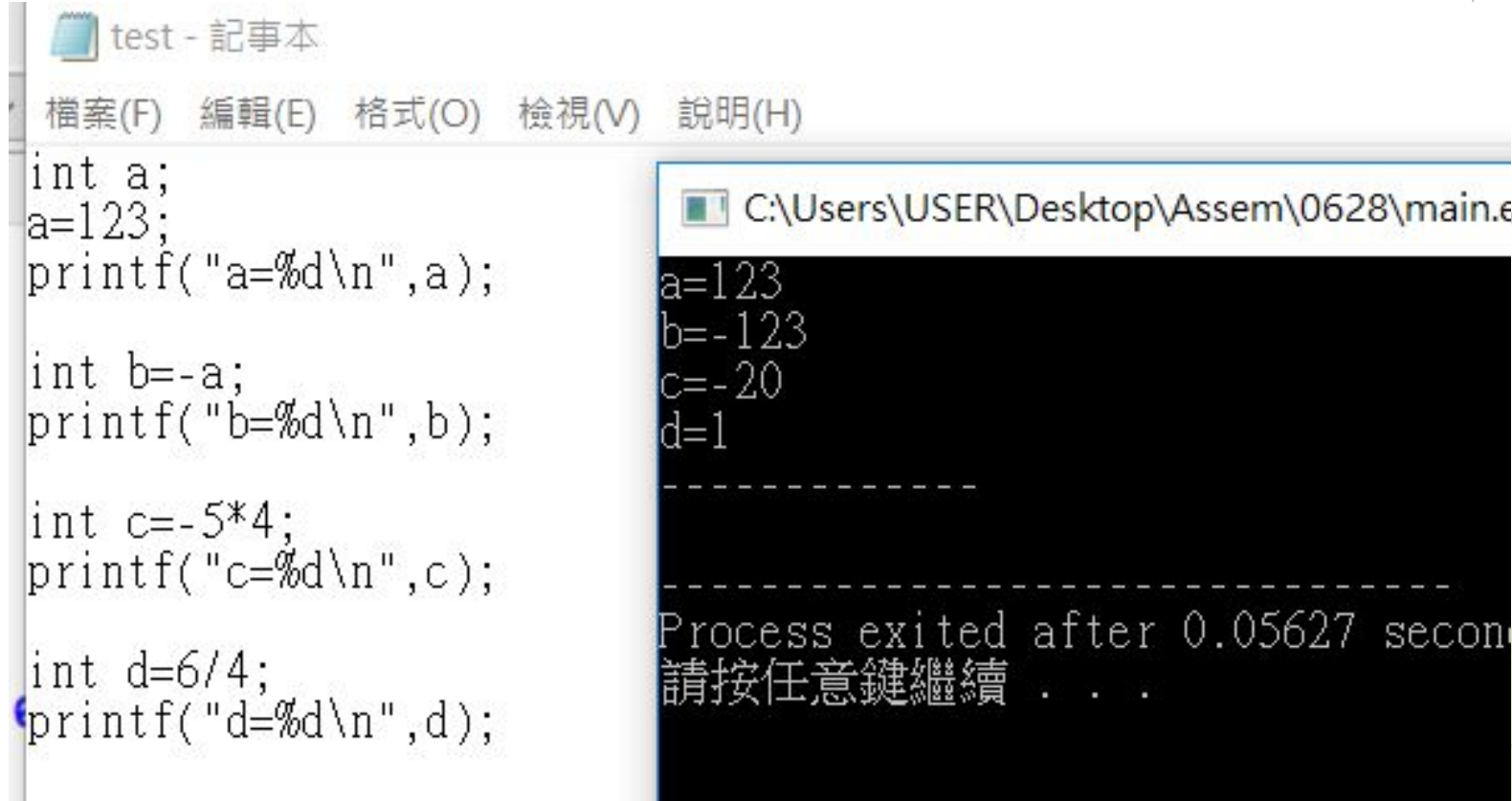
```
return;
```

```
}
```

```
variable.insert(pair<string,int>(key,value));
```

宣告變數

- ▶ 限制: 沒有區域變數, 迴圈內不能宣告 int



The image shows a Windows Notepad window titled "test - 記事本" containing a C program. The program defines four integer variables: 'a' (123), 'b' (-123), 'c' (-20), and 'd' (1). It uses printf to output each variable's value. Below the code, a black console window shows the program's execution output, which matches the code's printf statements. The console window title is "C:\Users\USER\Desktop\Assem\0628\main.e". After the output, it shows a separator of dashes, the message "Process exited after 0.05627 second", and a prompt in Chinese "請按任意鍵繼續 . . .".

```
int a;  
a=123;  
printf("a=%d\n",a);  
  
int b=-a;  
printf("b=%d\n",b);  
  
int c=-5*4;  
printf("c=%d\n",c);  
  
int d=6/4;  
printf("d=%d\n",d);
```

C:\Users\USER\Desktop\Assem\0628\main.e
a=123
b=-123
c=-20
d=1

Process exited after 0.05627 second
請按任意鍵繼續 . . .

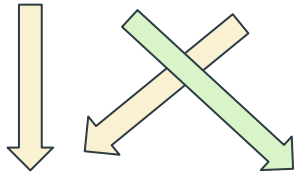
printf



printf_func(string str, bool assem)

► printf("OK%d %d123\n" , a , 12);

compute_int(int_fragment[++int_num]);



抓出 %d 的數量

str_fragment

- int int_num=-1;
for(int j=0; j<output_fragment.size(); j++){
 if(output_fragment[j]=="%d") cout<<compute_int(int_fragment[++int_num]);
 else cout<<output_fragment[j];
}
- 限制: 只有 %d 功能, 不支援 %s...

test - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

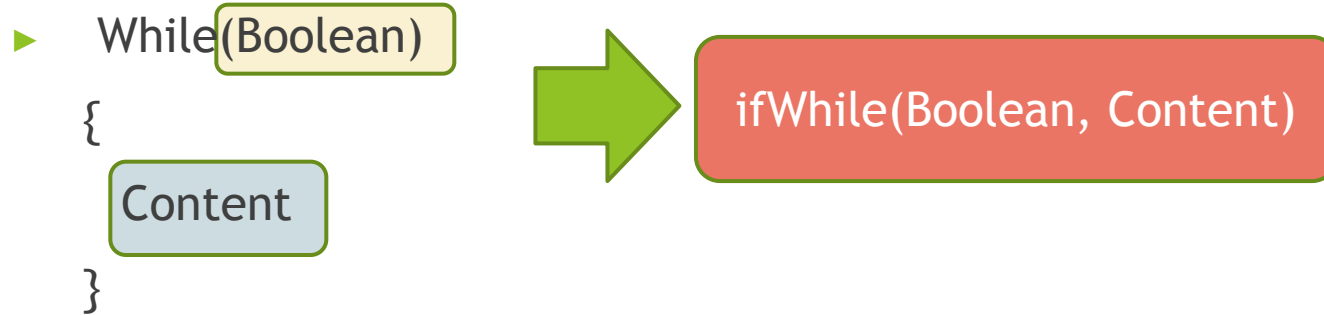
```
printf("hello\n");  
printf("ok\tok\n");  
  
int a=10;  
printf("a = %d\n",a);
```

C:\Users\USER\Desktop\

```
hello  
ok      ok  
a = 10  
-----
```


While

- ▶ 當exeFile 判斷第一個關鍵字為 while 時, 抓出Boolean, 抓出content



進入ifWhile後, 呼叫computebool, 判斷是否達成條件, 若達成會把Content丟進exeFile中執行

```
while (computebool ( Boolean) )  
{  
  exeFile(Content)  
}  
return;
```

test - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
int a=20;
int b=0;
while(a>10){
    while(b<=10) b++;
    a--;
    printf("a = %d\n",a);
}
```

C:\Users\USER\Desktop\Asse

```
a = 19
a = 18
a = 17
a = 16
a = 15
a = 14
a = 13
a = 12
a = 11
a = 10
-----
```

bool computebool (string Boolean)

左值 A

邏輯

右值 B

1.把左值A(string)，抓出後丟入compute_int，得int 型態的回傳值，用int a來暫存

2.判斷是否還有邏輯與右值

→沒有 → 若 $a \neq 0$ ，回傳true，否則回傳false

→有：

判斷邏輯，用int state作為代碼去儲存狀態($>$ 、 $<$ 、 $==$ 、 $>=$ 、 $<=$)

把右值B(string)，抓出後丟入compute_int，得int 型態的回傳值，用int b來暫存

左右值由compute_int做計算，可以是運算式

最後分case判斷，回傳布林值

If-else if -else

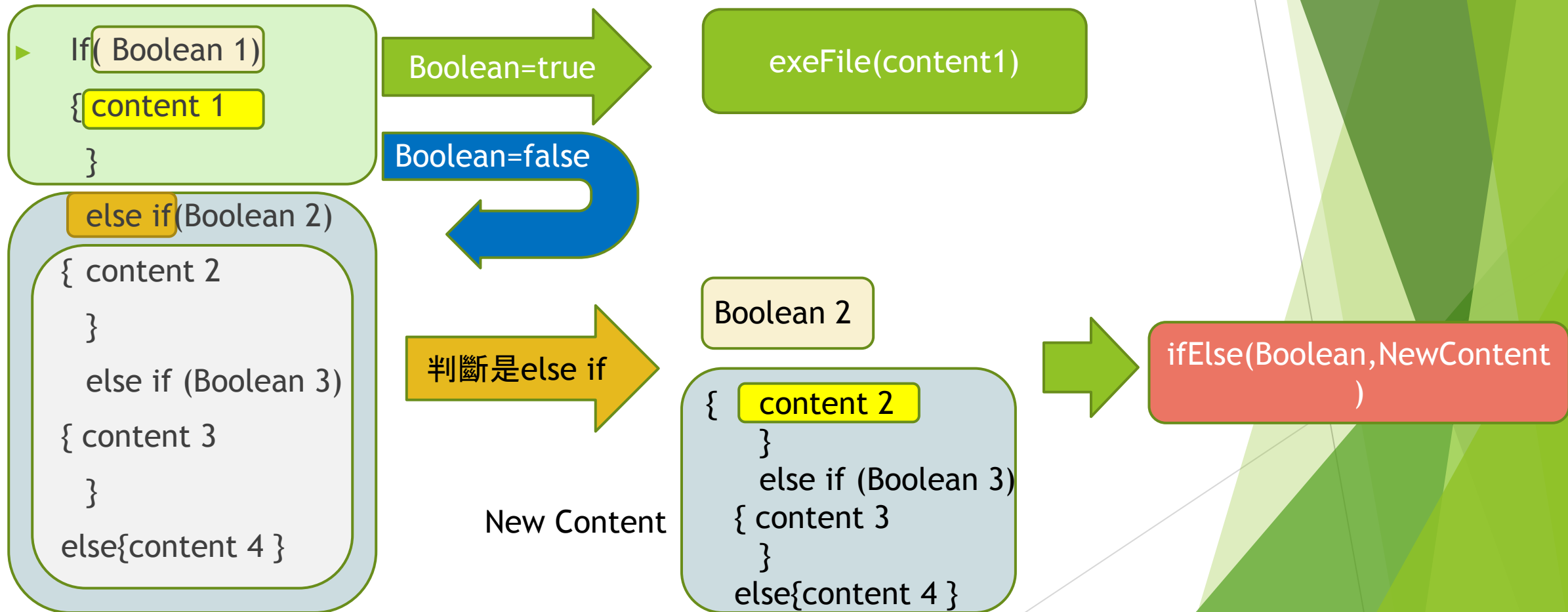
- ▶ 當exeFile 判斷第一個關鍵字為 if 時, 抓出Boolean, 把整組if else 都放進content
- ▶ If(Boolean 1)

```
{ content 1  
}  
else if(Boolean 2)  
{ content 2  
}  
else if (Boolean 3)  
{ content 3  
}  
else{content 4 }
```

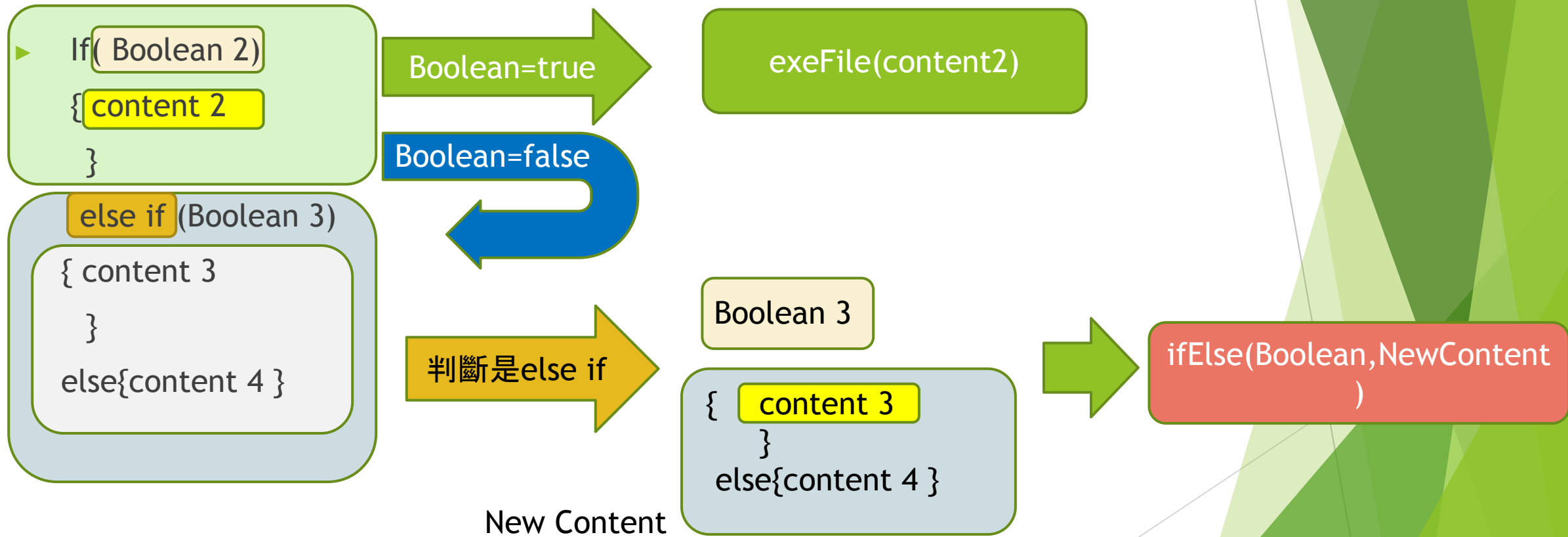


```
ifElse(Boolean, Content)
```

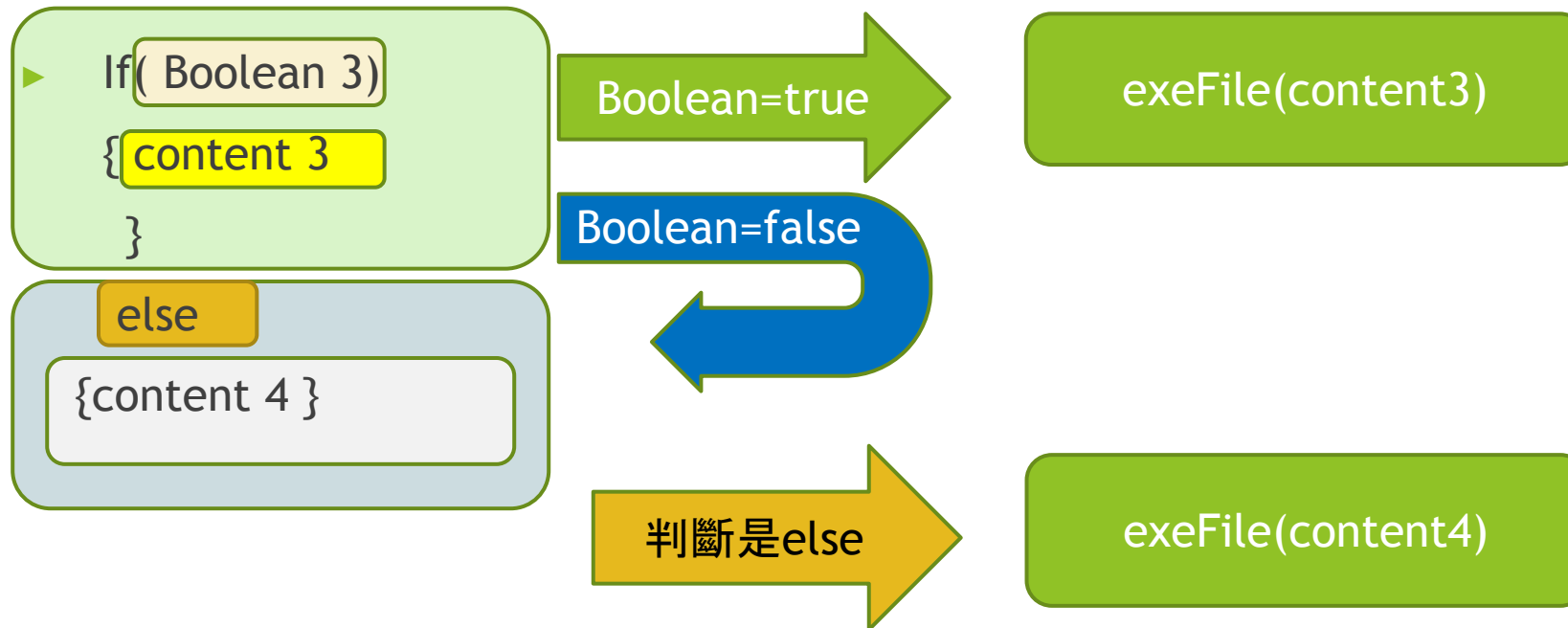
If-else if -else



If-else if -else



If-else if -else



test - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
for(int i=0; i<10; i++){  
    if(i<5){  
        if(i==1) printf("%d\n",i);  
        else if(i>3) printf("333\n");  
        else printf("okok\n");  
    }else if(i<=8){  
        printf("else if %d\n",i);  
    }else printf("else %d\n",i);  
}
```

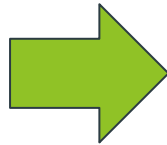
C:\Users\USER\De

```
okok  
1  
okok  
okok  
333  
else if 5  
else if 6  
else if 7  
else if 8  
else 9  
-----
```


for

- ▶ 當 exeFile 判斷第一個關鍵字為 for 時, 抓出 (declare; Boolean; compute) 以及抓出 content

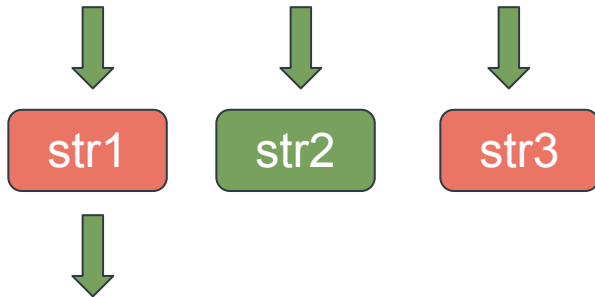
```
for(declare; Boolean; compute)
{
  Content
}
```



for_func(string boolean, string content,
bool assem)

- ▶ 進入 for_func() 後, 先將黃色區塊斷成 str1, str2, str3

```
for(declare; Boolean; compute)
```

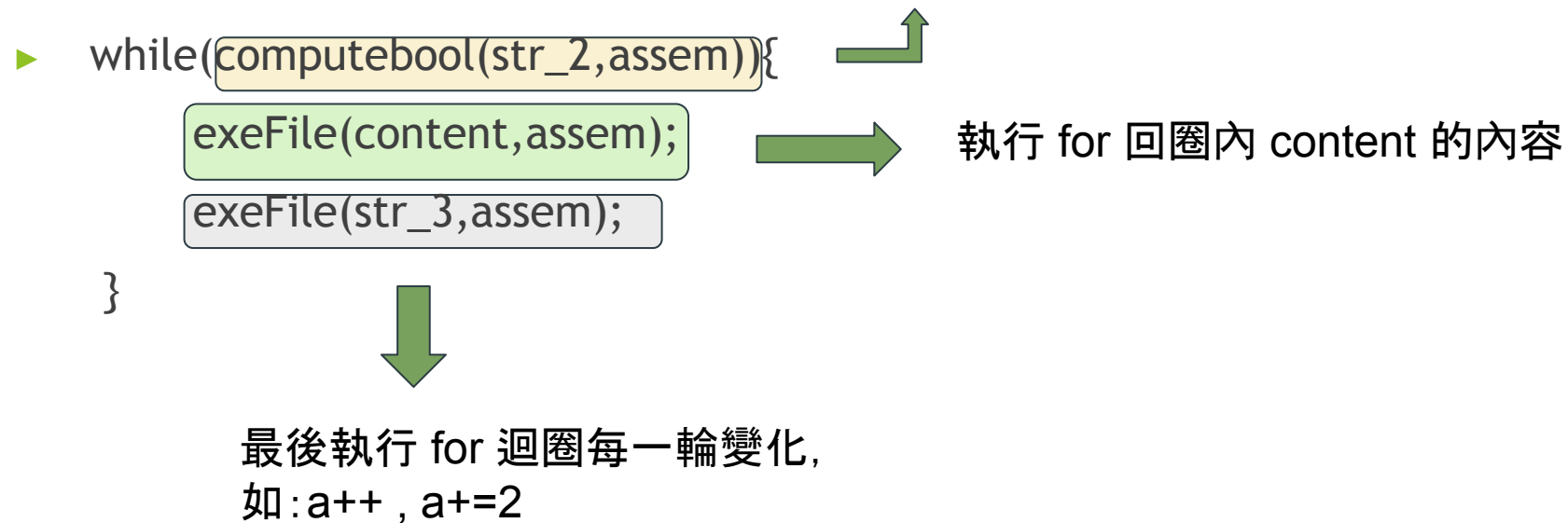


```
exeFile(str_1,assem);
```

執行 int a=0; 或是 a=0;

for_func(string boolean, string content, bool assem)

先判斷執行 for 迴圈的條件是否成立，當 $a < 10$ 等 boolean 值正確時，就會執行 for 迴圈內 content 的內容



test - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
int j=5;
for(int i=0; i<5; i++){
    for(j=3; j>0; j--) printf("j = %d\n", j);
    printf("i = %d\n", i);
    printf("=====\n");
}
```

C:\Users\USER\Desktop\Assem\0628

```
j = 3
j = 2
j = 1
i = 0
=====
j = 3
j = 2
j = 1
i = 1
=====
j = 3
j = 2
j = 1
i = 2
=====
j = 3
j = 2
j = 1
i = 3
=====
j = 3
j = 2
j = 1
i = 4
=====
```

Other 與 updateValue

- ▶ 當exeFile 判斷第一個關鍵字非「int」、「if」、「while」、「for」、「print」
- ▶ 進入Other後，判斷第一個字是否為已定義的變數
- ▶ 否→compile錯誤
- ▶ 是→呼叫updateValue

進入updateValue後，判斷：

變數

邏輯

值

判斷邏輯，用int state作為代碼去儲存狀態(+ 、-、*、/、+= 、-=、/=、*=、++、--)

跟數值做運算，存回變數裡

數值由compute_int做計算，可以是運算式

特殊困難- if for while 沒有大括號 不知道content要抓到哪裡

► if(Boolean)

```
for( ; Boolean ; )  
    while(Boolean)  
        a++;
```

else if(Boolean)

```
if(Boolean)  
    printf("content");
```

else

```
int c=3;
```

解決: 在if for while後判斷是否有大括號
沒有→判斷第一個key word是否為
if for while 之一, 若不是就抓到分號為止
若是則繼續遞迴判斷

Compute_int Outline

```
int compute_int(string str){  
  
    int len=str.length(); //Array.length()  
    char inorder[len+10], postorder[len]; //inorder來承接傳入的運算式  
                                           //postorder來承接轉換後的運算式  
    strcpy(inorder, str.c_str()); //將string轉為char，方便push pop  
    clearblank(inorder); //將傳入的運算式的空白去除  
  
    vector<string> res = to_postorder(inorder, postorder); //將處理好的運算式丟入function中  
    return calculate(res);  
}
```

clear



In->post

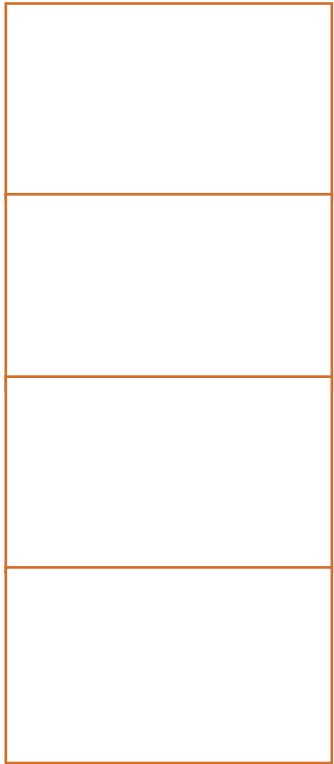


calculate

ClearBlank

```
for(int k=0 ; k<strlen(inorder) ; k++)  
    if(inorder[k]!=' ' && inorder[k]!='\n' && inorder[k]!='\t' &&inorder[k]!='\0')  
        inorder[n++] = inorder[k];
```

In->post



Operator Stack

Case ')' push

Case ')' 尋找 '(' 並將裡面的運算符號pop

Case '+' '*' '/' 先檢查運算優先度, 再決定要pop還是push

Case '-' 需考慮是否為減還是負

Solution: 直接push 0 進去, 可在組語中看見
也因此操作時, 負數需加()



Postorder

In->post

需判斷;

1. 是否為變數 (A_) (_A) (A123) (A~z)

Solution:

```
while(IsDight(inorder[i]) || (inorder[i]>='A' && inorder[i]<='Z'))  
{  
    char tmpChar [] ={inorder[i],'\0'};  
    temp.append(tmpChar);  
    i++;
```


(片段)

2. 整數時, 也必須抓取個位數以上

Soltion:

```
while(IsDight(inorder[i])){  
    value=value*10+(inorder[i]-48);  
    i++;
```

Calculate(function)

```
if(!(ssTest4>>a) ){
    int temp = call_variable(sa);
    if(temp!=-1){
        a=temp;
    }
    else{
        cout << "in sa err" <<endl;
    }
}
```

BUG

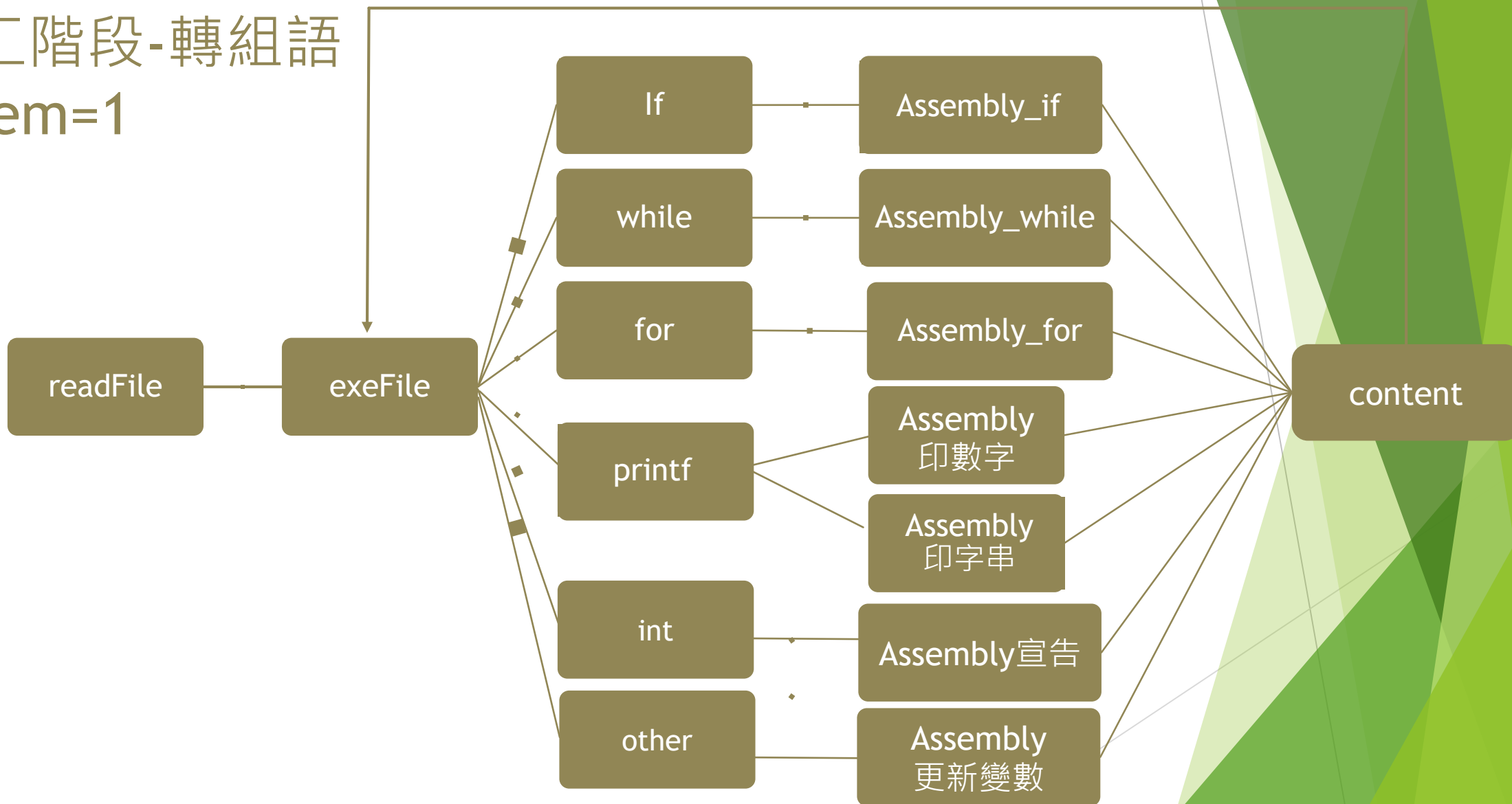
```
stringstream ssTest;
ssTest<<sb;
if(!(ssTest>>b) ){
    //cout << "b="<<b <<endl;
    int temp = call_variable(sb);
    if(temp!=-1){
        cout << sb << "=" << temp << endl;
        b=temp;
    }
}
```

```
stringstream ssTest3;
string sstemp;
ssTest3<<a;
ssTest3>>sstemp;
cal_string.push_back(sstemp);
```

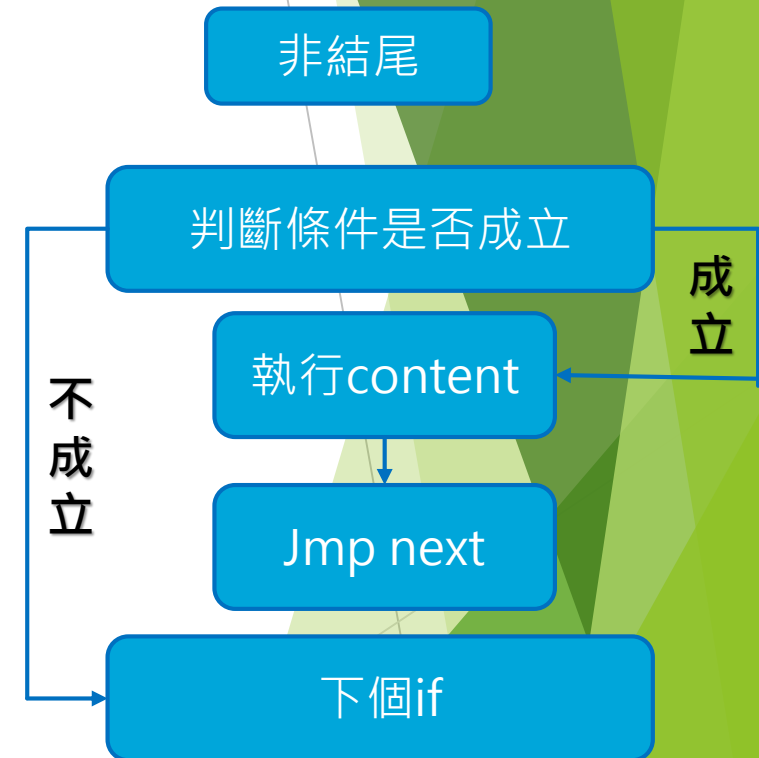
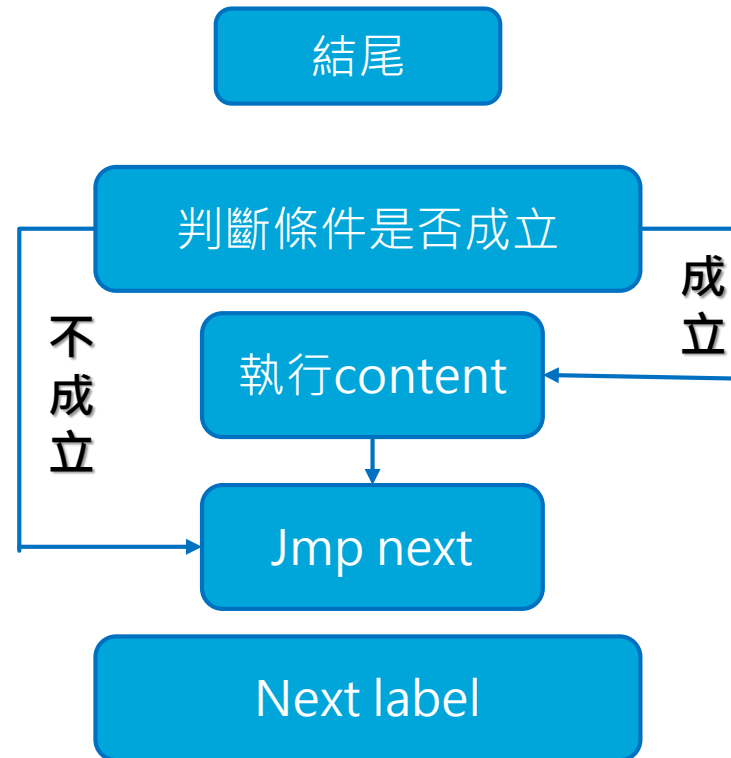
程式架構

第二階段-轉組語

assem=1



assembly_if



A1:

```
B1:
...
jmp Bnext
B2:
...
jmpBnext
Bnext:
jmp Anext
```

A2:

```
C1:
...
jmp Cnext
C2:
...
jmpCnext
Cnext:
jmp Anext
```

Anext:

assembly_for

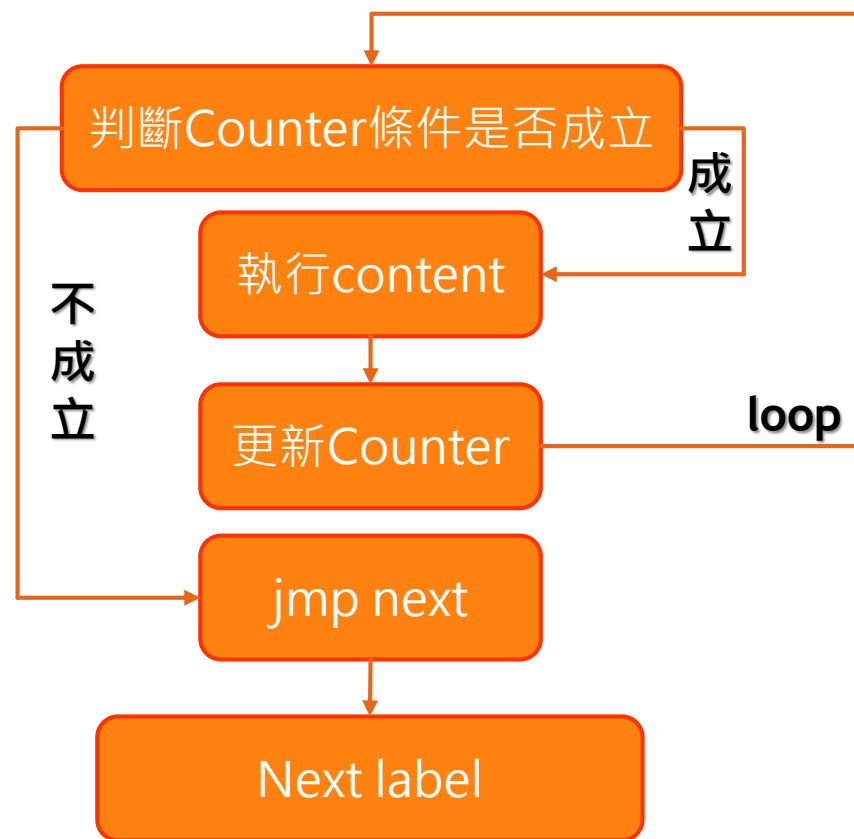
Counter
宣告

Counter
判別式

Counter
運算

內容

Loop編號



LOOP1:

...
jmp LOOP1next

LOOP2:

...
jmp LOOP2next
content
update Counter
jmp LOOP2
LOOP2next:

update Counter
jmp LOOP1

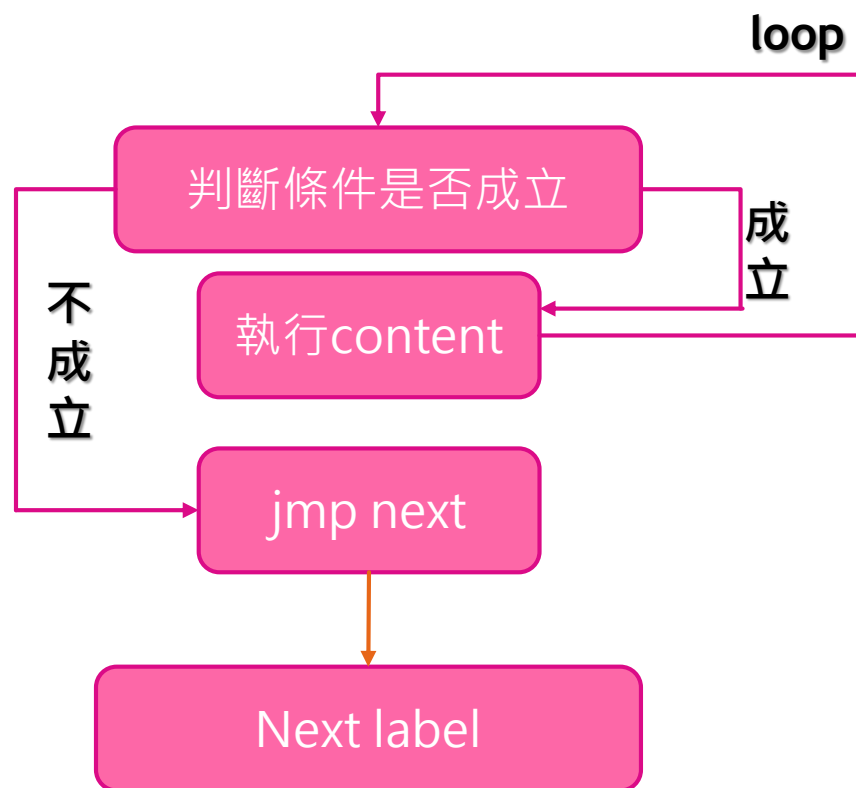
LOOP1next:

assembly_while

判斷式

內容

While編號



assembly_int 、 assembly_updatevalue

變數名稱

初始值

變數名稱

運算子

運算元

Name DWORD value

跟C++的一樣OYO

assembly_computebool

- ▶ 雙運算元boolean:
 - ▶ 算出兩個運算元的值
 - ▶ `cmp` 兩個運算元
 - ▶ 依傳入state產生conditional jump
- ▶ 單一運算元boolean:
 - ▶ 算出該運算元的值
 - ▶ 與0比較
 - ▶ `jb`

assembly_printf_str、 assembly_printf_int

- ▶ 印String
 - ▶ 將String依char形式依序印出
- ▶ 印int
 - ▶ 判別要印出的值為正或負
 - ▶ 正值call WriteDec
 - ▶ 負值call WriteInt