APCS專題講座

饒建奇 副教授

淡江大學電機工程學系

2020/03/16@中和高中

饒建奇

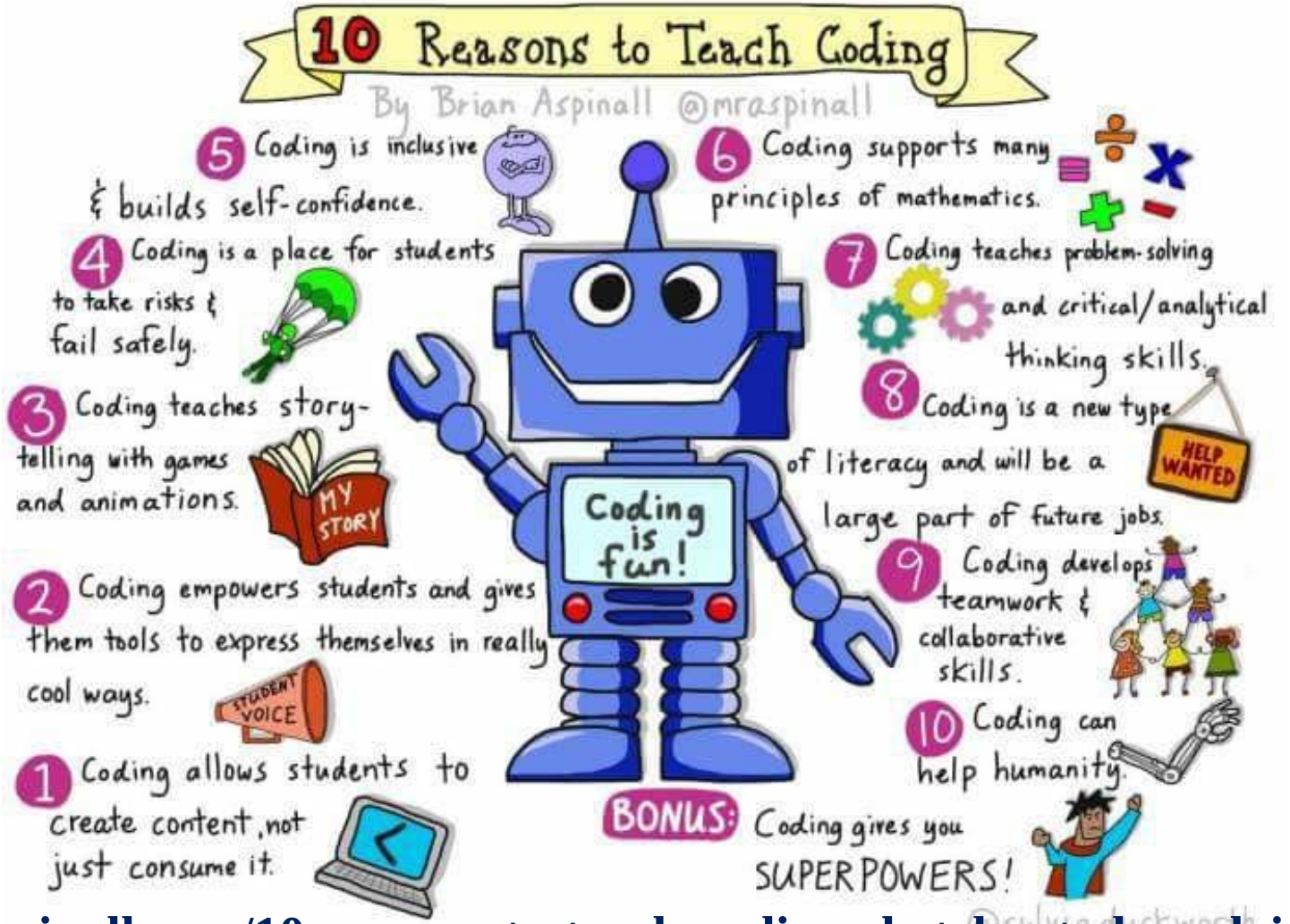
Outline

- 程式教育的重要性
- APCS是什麼
- C/C++基本介紹

Outline

- 程式教育的重要性
- APCS是什麼
- C/C++基本介紹

Why Coding?* (1/2)



*https://brianaspinall.com/10-reasons-to-teach-coding-sketchnote-by-sylviaduckworth/

Why Coding? (2/2)

「引南港高中高慧君老師文」：

1. 建立自信、
2. 安全的冒險、
3. 通過遊戲和動畫講故事、
4. 给學生工具讓他们很酷的表達自己、
5. 創造内容而不是消費、
6. 程式教育可學到許多數学規則、
7. 程式設計教會學生問題的解決方式和批判性思维方式與技能、
8. 程式教育是一種新文化並且會成為未来工作的重要组成部分、
9. 程式設計鍛練團隊合作技能、
10. 程式設計可以修煉人性。

全民寫程式

比爾蓋茲認為程式教育是
「每個學生都應該學習的21世紀的基本技能」

- 愛沙尼亞從2012年
- 英國從2014年
- 美國前總統歐巴馬於2016年提出「全民電腦科學倡議」
- 日本從2017年
- 台灣的108(2019)課綱納入程式設計課程

學習程式設計的重要性

- 創造能力
- 自學能力
- 邏輯思考能力
- 運算思維能力

➢解決問題的能力

十二年國教推動程式教育的原因 (1/2)

- 程式設計在資訊科技中扮演著基礎並重要的角色
  - 程式設計
  - 演算法
  - 系統平台
  - 資料表示、處理與分析
  - 資訊科技應用
  - 資訊科技與人類社會

十二年國教推動程式教育的原因 (2/2)

- 學生透過撰寫程式能夠實驗課堂中學習到的理論並發揮自己的創意寫出各式各樣功能的軟體
  - 運算思維與問題解決
  - 資訊科技與合作共創
  - 資訊科技與溝通表達
  - 資訊科技的使用態度

Outline

- 程式教育的重要性
- APCS是什麼
  - 資料來源：https://apcs.csie.ntnu.edu.tw/
- C/C++基本介紹

何謂APCS*

- Advanced Placement Computer Science
  - 大學程式設計先修檢測：評量學生程式設計的能力
  - 其檢測模式乃參考美國大學先修課程（Advanced Placement，AP），與各大學合作命題，並確定檢定用題目經過信效度考驗，以確保檢定結果之公信力。

*https://apcs.csie.ntnu.edu.tw/

推動緣由與活動

- 如今學生的資訊能力日益受到重視，但資訊科學並不在學測考試項目中，不論在推薦入學、申請入學或考試入學等入學管道，對於學生的資訊能力尚缺乏客觀的評量依據…
- 每年2月、6月及10月各一次

檢測內容

- 包含「程式設計觀念題」及「程式設計實作題」兩科目，於電腦教室試場進行線上測驗。

科目	說明
程式設計觀念題	兩份測驗題本共計40個試題，分兩節次施測
程式設計實作題	一份測驗題本共計4個題組

評分方式

- 程式設計觀念題為選擇題，分兩節次檢測，檢測分數為合併計分，滿分100分；
- 程式設計實作題為單節次檢測，以撰寫完整程式或副程式計分，滿分400分；
- 兩科目均採取自動評分與統計，評分過程不涉及主觀因素。

程式設計觀念題

- 單選題(含題組)，以運算思維、問題解決與程式設計概念測試為主。
- 測驗題型
  - 程式運行追蹤(code tracing)
  - 程式填空(code completion)
  - 程式除錯(code debugging)
  - 程式效能分析(code performance analysis)
  - 基礎觀念理解 (basic concepts understanding)
- 題目若需提供程式片段，則以 C 語言命題。

程式設計觀念題範圍 (1/2)

- 程式設計基本觀念(basic programming concepts)
- 資料型態(data types)，常數(constants)，變數(variables)，視域(scope)：全域(global) / 區域(local)
- 控制結構(control structures)
- 迴路結構(loop structures)

程式設計觀念題範圍 (2/2)

- 函式(functions)
- 遞迴(recursion)
- 陣列與結構(arrays and structures)
- 基礎資料結構(basic data structures)，包括：佇列(queues)和堆疊(stacks)
- 基礎演算法(basic algorithms)，包括：排序(sorting) 和搜尋(searching)

觀念題範例

經過運算後，右側程式的輸出為何?

(A) 1275
(B) 20
(C) 1000
(D) 810

```
for (i=1; i<=100; i=i+1) {
  b[i] = i;
}

a[0] = 0;
for (i=1; i<=100; i=i+1) {
  a[i] = b[i] + a[i-1];
}
printf ("%d\n",  a[50]-a[30]);
```

程式設計實作題

- 以撰寫完整程式或副程式為主
- 可自行選擇以 C, C++, Java, Python 撰寫程式

程式設計實作題範圍 (1/2)

- 輸入與輸出 (input and output)
- 算術運算(arithmetic operation)，邏輯運算 (logical operation)，位元運算 (bitwise operation)
- 條件判斷與迴路 (conditional expressions and loop)
- 陣列與結構(arrays and structures)

程式設計實作題範圍 (2/2)

- 字元 (character)，字串 (string)
- 函數呼叫與遞迴 (function call and recursion)
- 基礎資料結構 (basic data structures)，包括：佇列 (queues)，堆疊 (stacks)，樹狀圖 (tree)，圖形 (graph)
- 基礎演算法 (basic algorithms)，包括：排序 (sorting)，搜尋 (searching)，貪心法則 (greedy method) ，動態規劃 (dynamic programming)

實作題範例

一次考試中，於所有及格學生中獲取最低分數者最為幸運，反之，於所有不及格同學中，獲取最高分數者，可以說是最為不幸，而此二種分數，可以視為成績指標。 請你設計一支程式，讀入全班成績(人數不固定)，請對所有分數進行排序，並分別找出不及格中最高分數，以及及格中最低分數。 當找不到最低及格分數，表示對於本次考試而言，這是一個不幸之班級，此時請你印出：「worst case」；反之，當找不到最高不及格分數時，請你印出「best case」。
註：假設及格分數為60，每筆測資皆為0~100間整數，且筆數未定。

輸入範例一：
```
10
0 11 22 33 55 66 77 99 88 44
```

輸出範例一：
```
0 11 22 33 44 55 66 77 88 99
55
66
```

輸入範例二：
```
1
13
```

輸出範例二：
```
13
13
worst case
```

輸入範例三：
```
2
73 65
```

輸出範例三：
```
65 73
best case
65
```

檢測時程

科目	節次	入場時間	檢測起始	提前交卷 起始時間	提前交卷 截止時間	檢測結束
觀念題	第一節	09:30	09:40	10:10	10:30	10:40
	第二節	11:00	11:10	11:40	12:00	12:10
實作題	第三節	13:30	13:40	14:10	16:00	16:10

成績說明

級分	程式設計觀念題	程式設計實作題	
	分數範圍	分數範圍	能力說明
五	90~100	350~400	具備常見資料結構與基礎演算程序運用能力
四	70~89	250~349	具備程式設計與基礎資料結構運用能力
三	50~69	150~249	具備基礎程式設計與基礎資料結構運用能力
二	30~49	50~149	具備基礎程式設計能力
一	0~29	0~49	尚未具備基礎程式設計能力

最近兩次檢測成績統計(2018/5/10)

	程式設計觀念題						
程式設計實作題	級別	五	四	三	二	一	總人數
	五	18	9	0	0	0	27
	四	27	31	0	0	0	58
	三	41	133	23	3	1	201
	二	35	332	253	86	8	714
	一	14	203	419	549	349	1534
	總人數	135	708	695	638	358	2534

11%

33%

109學年度大學個人申請入學篩選門檻 (1/4)

學校	學系	程式設計觀念題	程式設計實作題
國立成功大學	資訊工程學系	5級分	4級分
國立中央大學	資訊工程學系	4級分	4級分
國立交通大學	資訊工程學系	4級分	4級分
國立清華大學	資訊工程學系	4級分	4級分
國立中興大學	資訊科學與工程學系	4級分	3級分
國立中興大學	資訊管理學系	4級分	3級分
國立彰化師範大學	資訊工程學系	4級分	3級分
國立成功大學	工業與資訊管理學系	4級分	3級分
國立政治大學	資訊科學系	4級分	3級分
國立臺北大學	資訊工程學系	4級分	3級分
國立臺灣師範大學	資訊工程學系	4級分	3級分

109學年度大學個人申請入學篩選門檻 (2/4)

學校	學系	程式設計觀念題	程式設計實作題
國立中央大學	資訊管理學系	3級分	3級分
國立暨南國際大學	資訊工程學系	3級分	3級分
國立暨南國際大學	資訊管理學系	3級分	3級分
國立臺北教育大學	數學暨資訊教育學系資訊組	3級分	3級分
國立臺灣海洋大學	資訊工程學系	3級分	3級分
元智大學	資訊工程學系	3級分	2級分
國立聯合大學	資訊管理學系	3級分	2級分
國立金門大學	資訊工程學系	3級分	2級分
國立高雄大學	資訊工程學系	3級分	2級分
國立高雄師範大學	軟體工程與管理學系	3級分	2級分
慈濟大學	醫學資訊學系	3級分	2級分
臺北市立大學	資訊科學系	3級分	2級分

109學年度大學個人申請入學篩選門檻 (3/4)

學校	學系	程式設計觀念題	程式設計實作題
國立聯合大學	資訊工程學系	2級分	2級分
東海大學	資訊工程學系人工智慧組	2級分	2級分
東海大學	資訊工程學系資電工程組	2級分	2級分
東海大學	資訊工程學系軟體工程組	2級分	2級分
淡江大學	資訊工程學系	2級分	2級分
淡江大學	電機工程學系電機資訊組	2級分	2級分
輔仁大學	資訊工程學系	2級分	2級分
逢甲大學	資訊工程學系	2級分	2級分
銘傳大學	資訊工程學系(桃園校區)	2級分	2級分

109學年度大學個人申請入學篩選門檻 (4/4)

學校	學系	程式設計觀念題	程式設計實作題
實踐大學	資訊科技與管理學系(臺北校區)	2級分	-
義守大學	資訊工程學系	2級分	-
長榮大學	資訊暨設計學院學士班	2級分	-
靜宜大學	資訊傳播工程學系	2級分	-
靜宜大學	資訊工程學系	2級分	-
靜宜大學	資訊管理學系	2級分	-

Outline

- 程式教育的重要性
- APCS是什麼
- C/C++基本介紹

Structured Programming

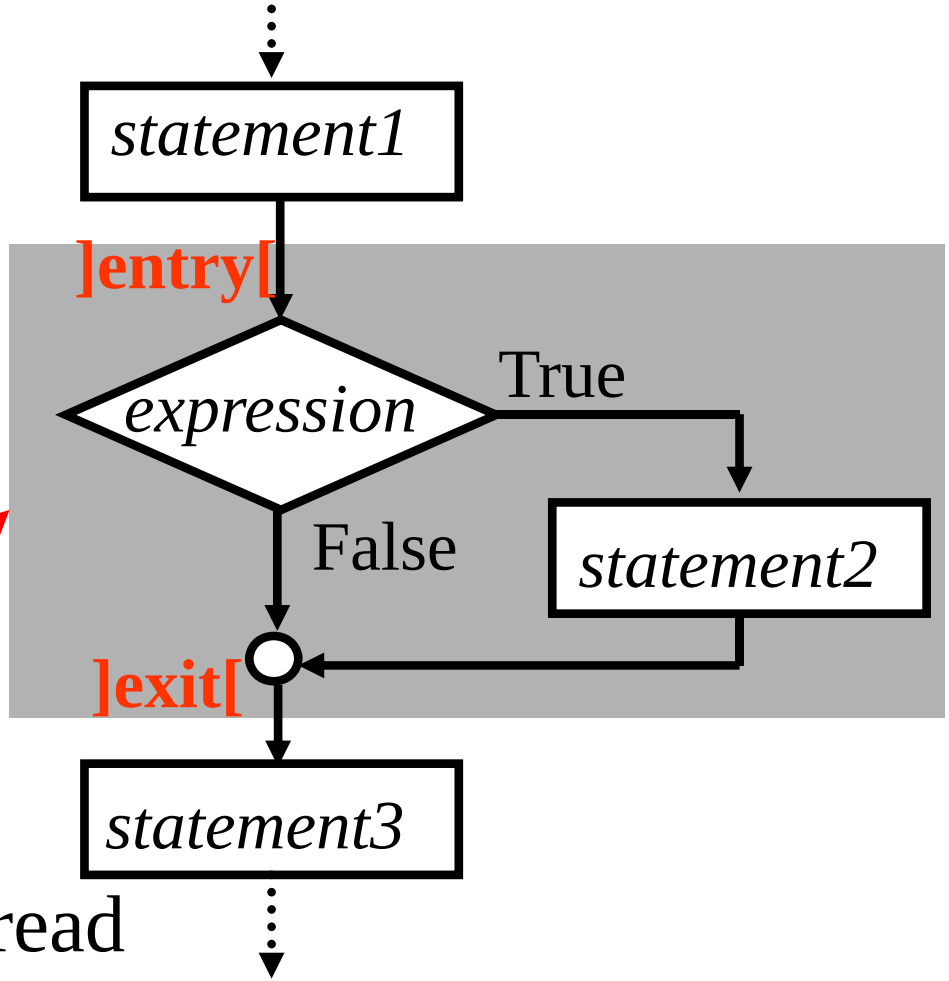
- A structured program should be written in only three control structures [Bohm and Jacopini]
  - **Sequence structure**
    - Built into C/C++-- statements are executed sequentially by default
  - **Selection structures**
    - C/C++ has three types-- **if**, **if/else**, and **switch**
  - **Repetition structures**
    - C/C++ has three types-- **while**, **do/while**, and **for**

Structured C/C++ Programs

- A structured C/C++ program can be constructed *merely from three control structures* resulted from the following statements
  - **sequence, if, if/else, switch, while, do/while, for**
- The above seven statements can be combined in only two ways
  - **Stacked**
  - **Nested**

The if Selection Structure

```
……
<statement1>
if (<expression>)
        <statement2>
<statement3>
……
```

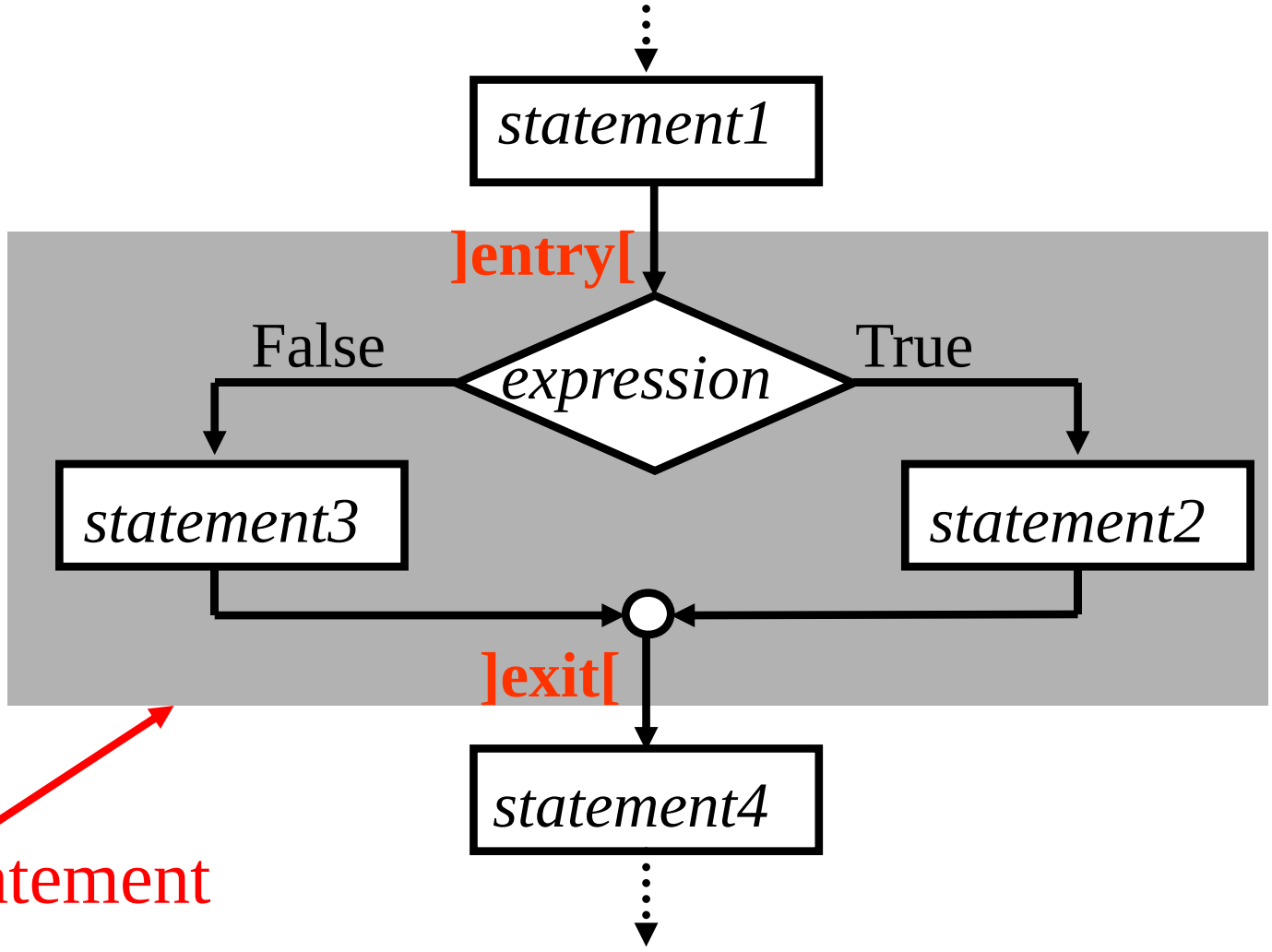


if statement

- Indenting makes programs easier to read
  - C/C++ compiler ignores whitespace characters (Blank, Tab, Enter)

The if/else Selection Structure

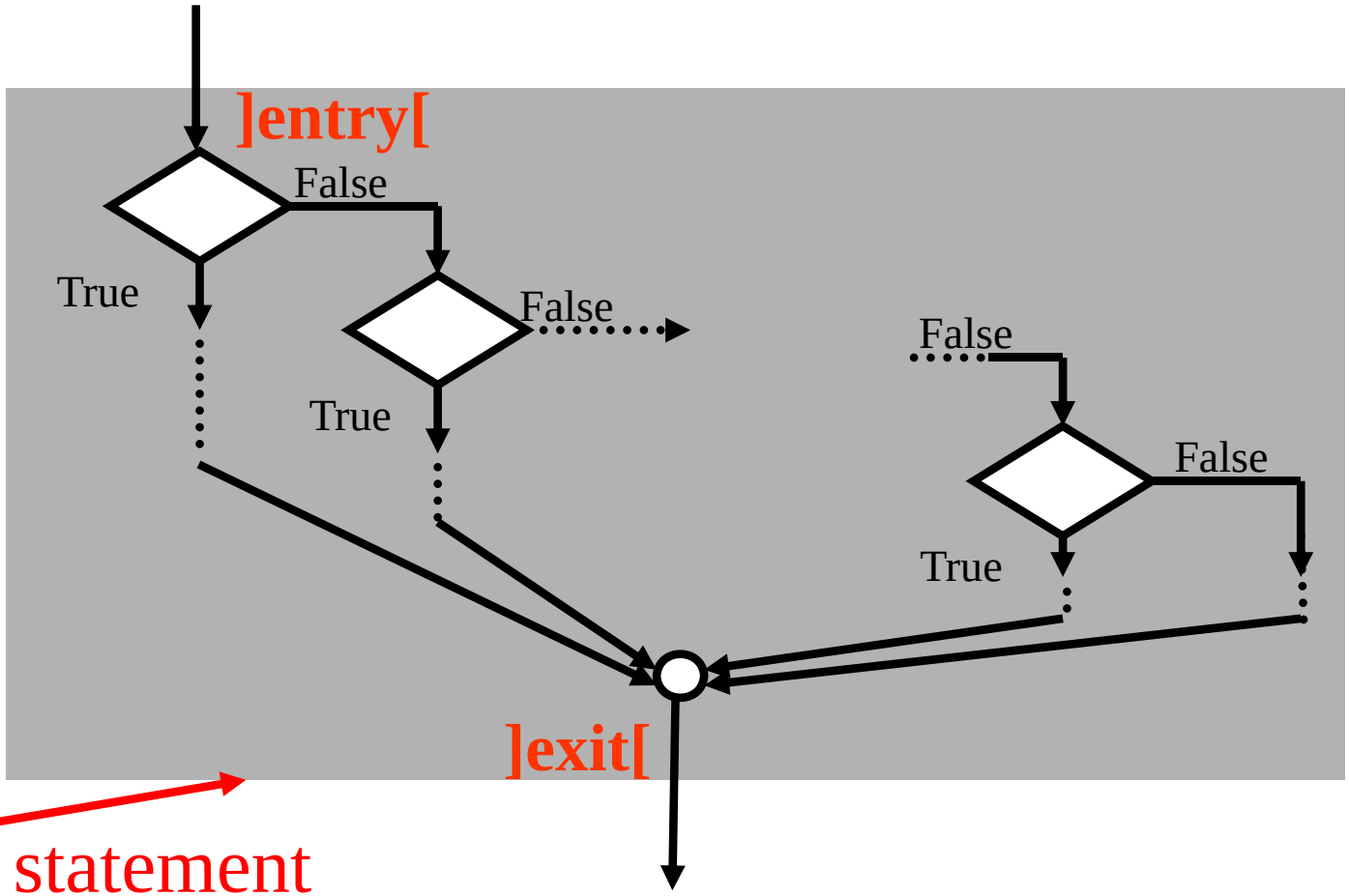
```
……
<statement1>
if (<expression>)
        <statement2>
else
        <statement3>
<statement4>
……
```



if/else statement

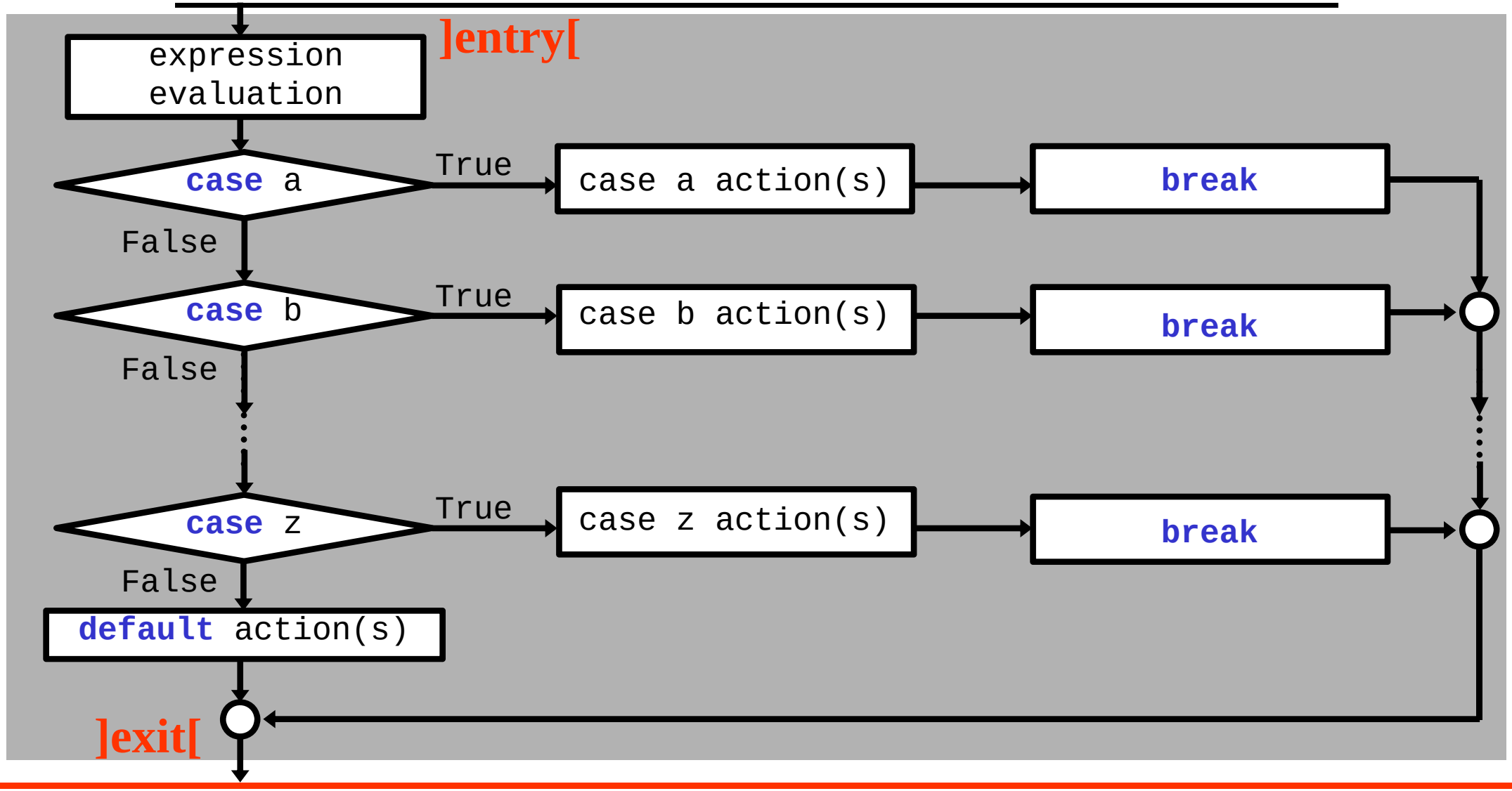
The if/else if Selection Structure

```
.....
if (<expression1>)
        <statement1>
[else if(<expression2_i>)
        <statement2_i>]*
[else
        <statement3>]
.....
```

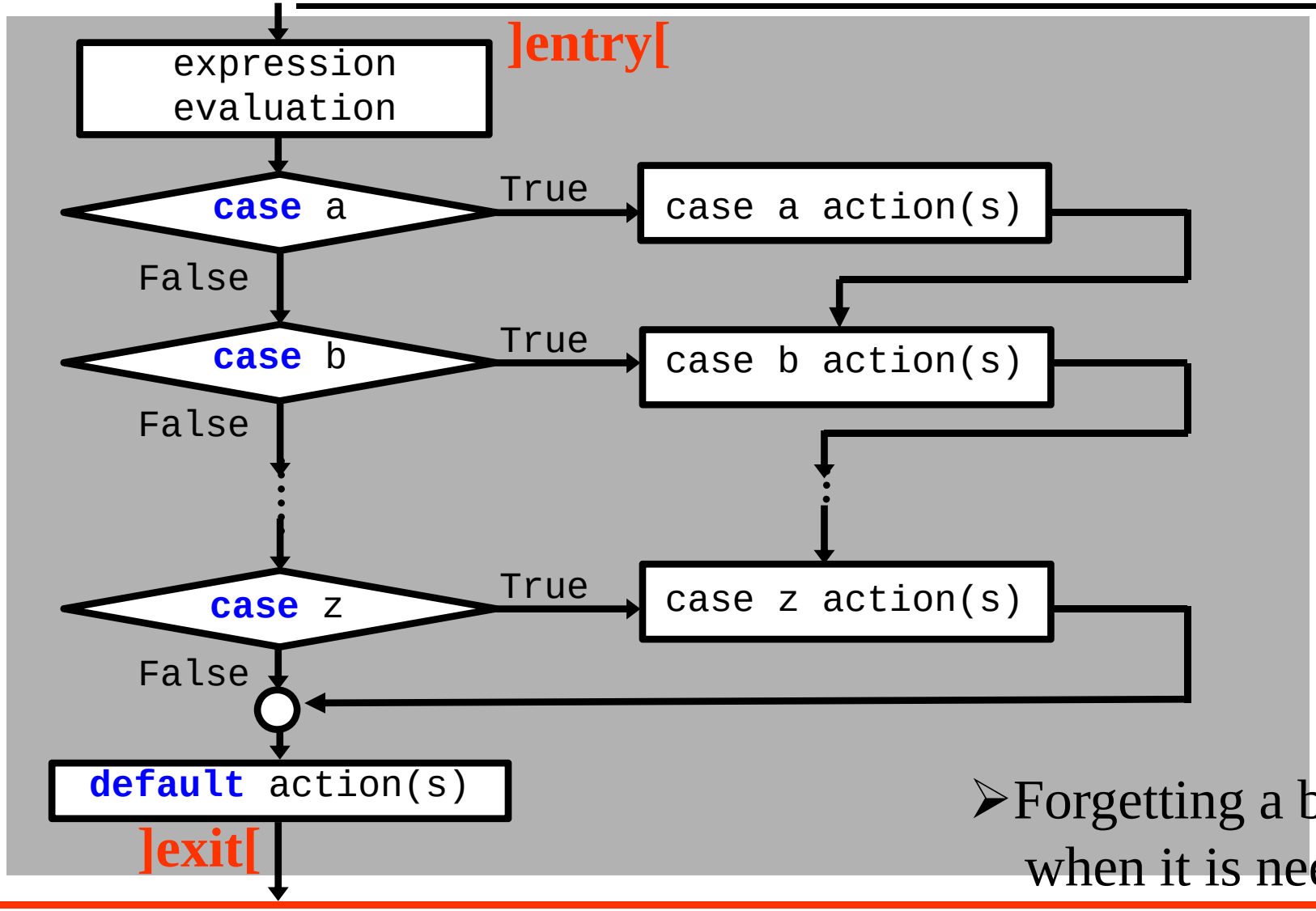


if/else if statement

The switch Selection Structure



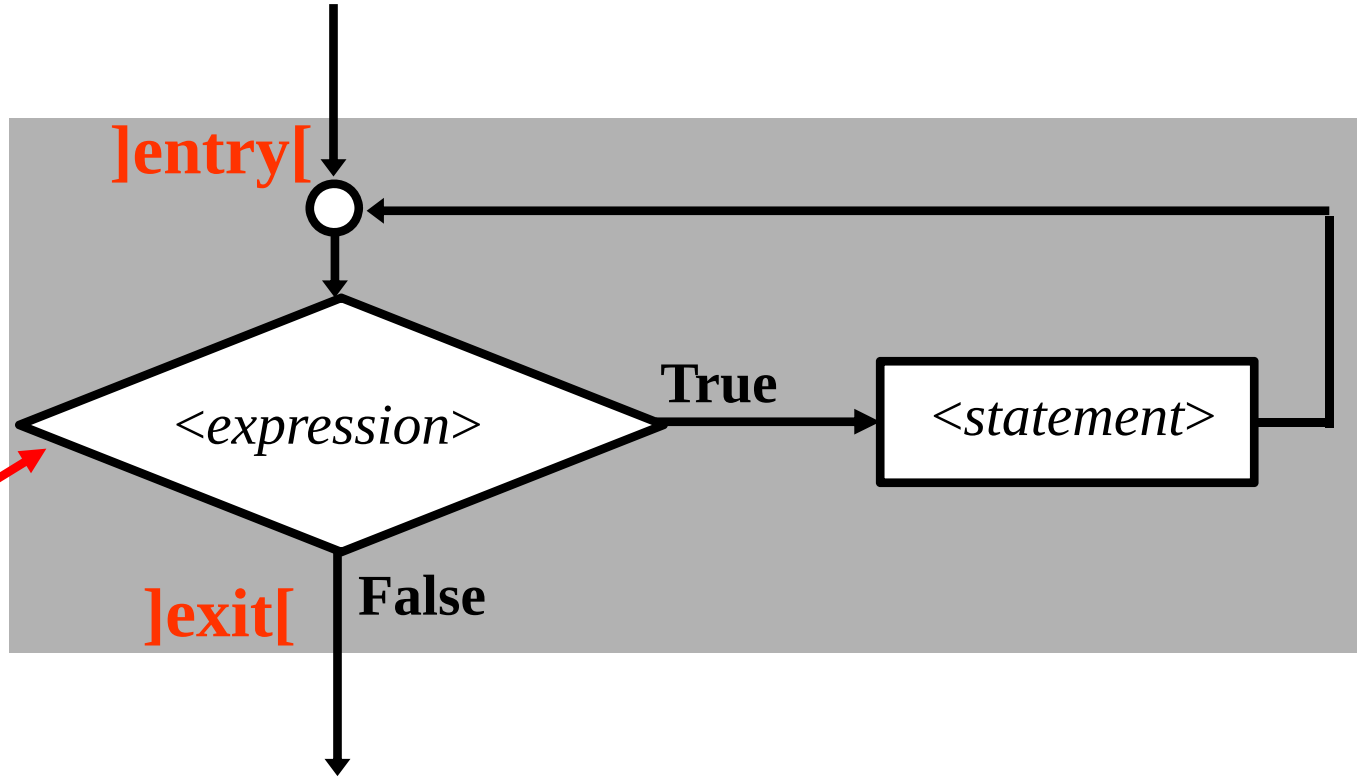
Without break Statements



- Forgetting a break statement when it is needed is a logic error

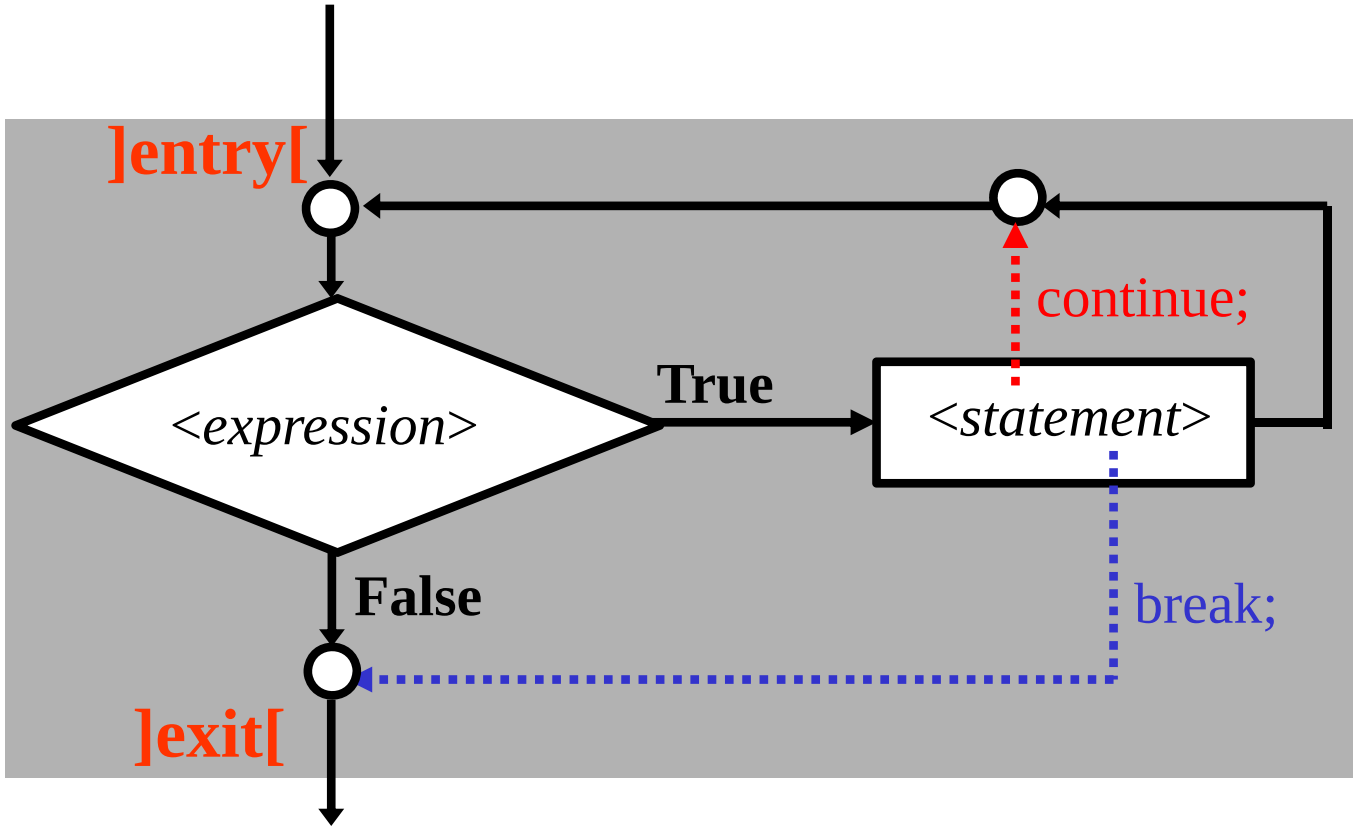
The while Repetition Structure

```
.....
while(<expression>)
        <statement>
.....
```



while-statement

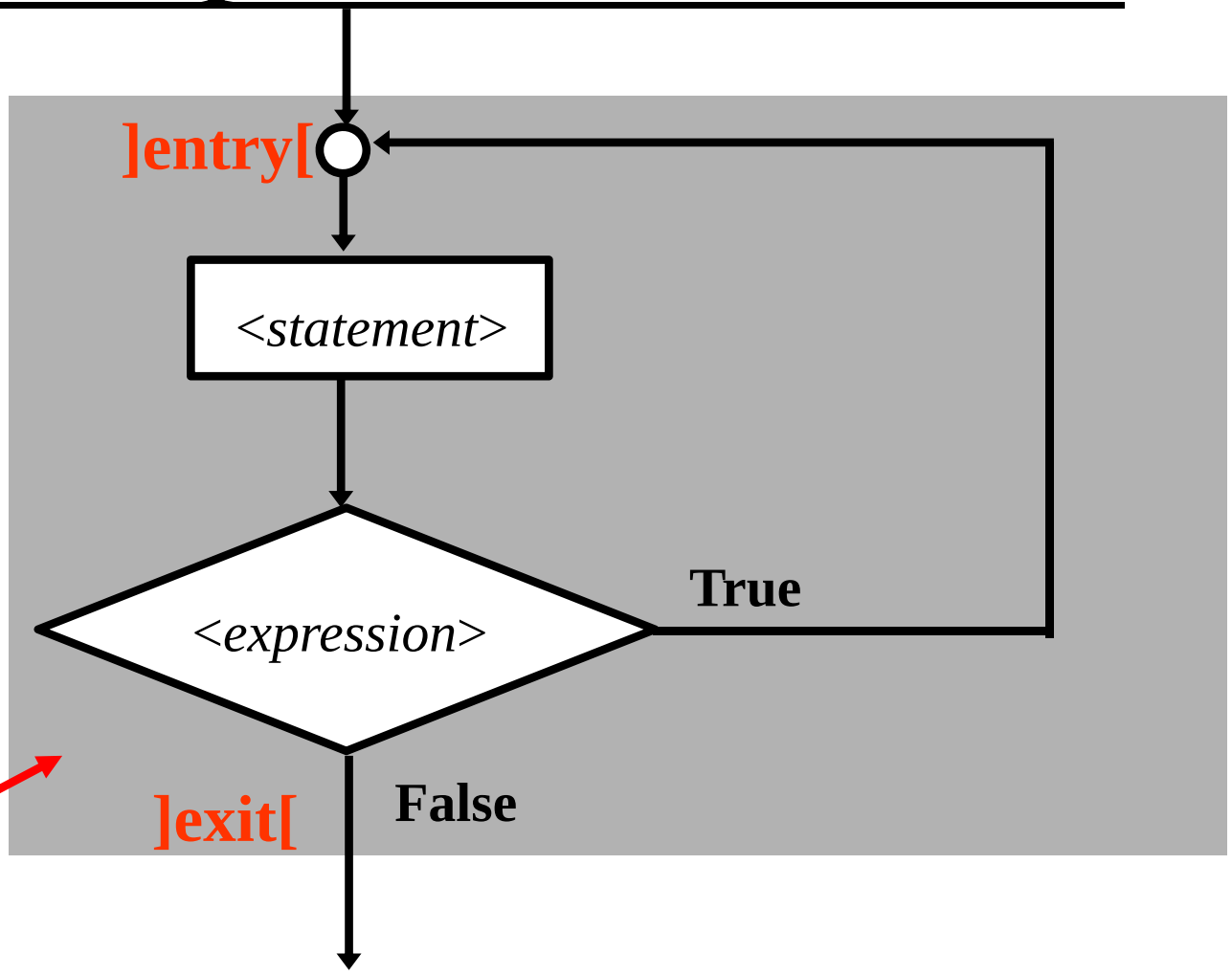
The while-break/continue Structure



The **while** loop with break/continue statement

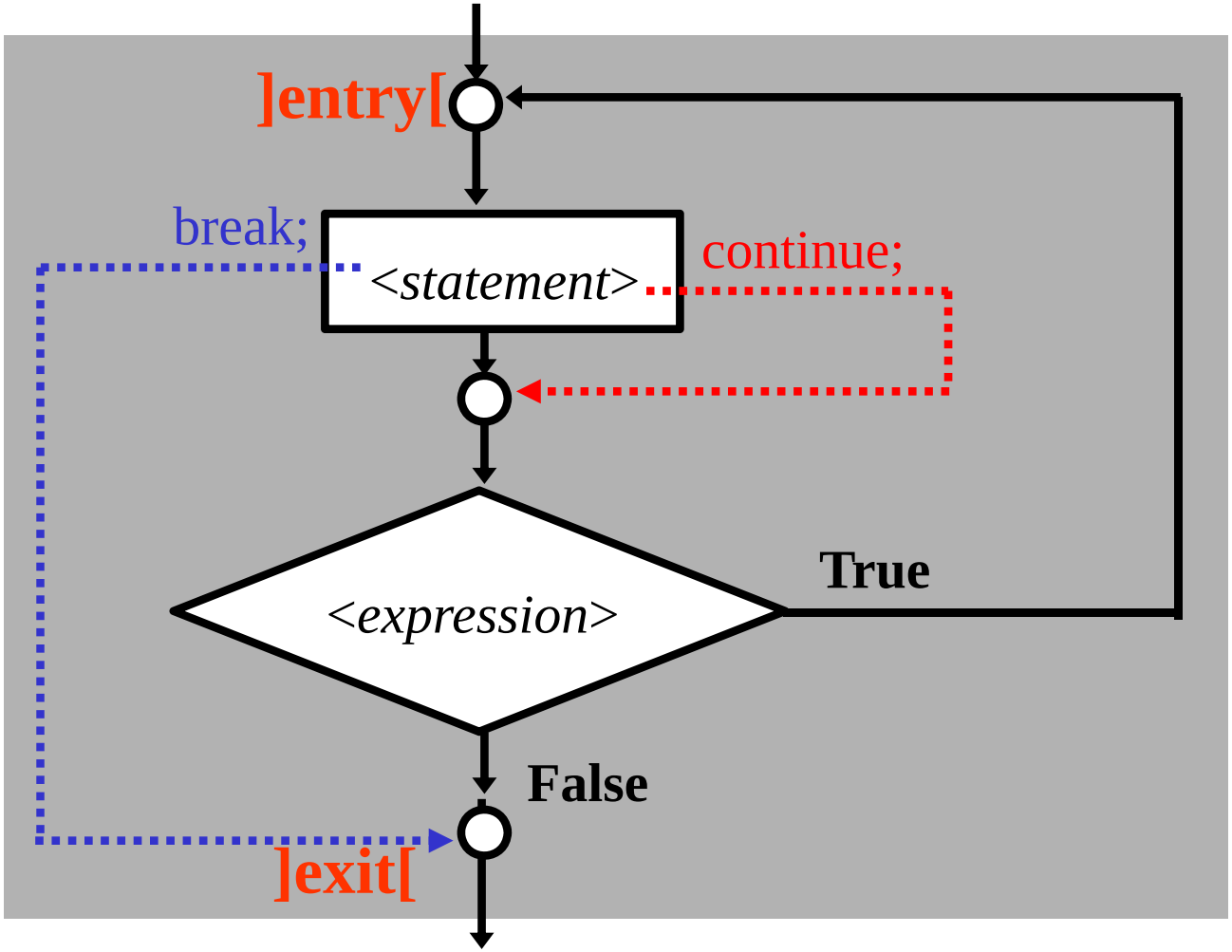
The do/while Repetition Structure

```
.....
do
        <statement>
while(<expression>);
.....
```



do/while-statement

The do/while-break/continue Structure



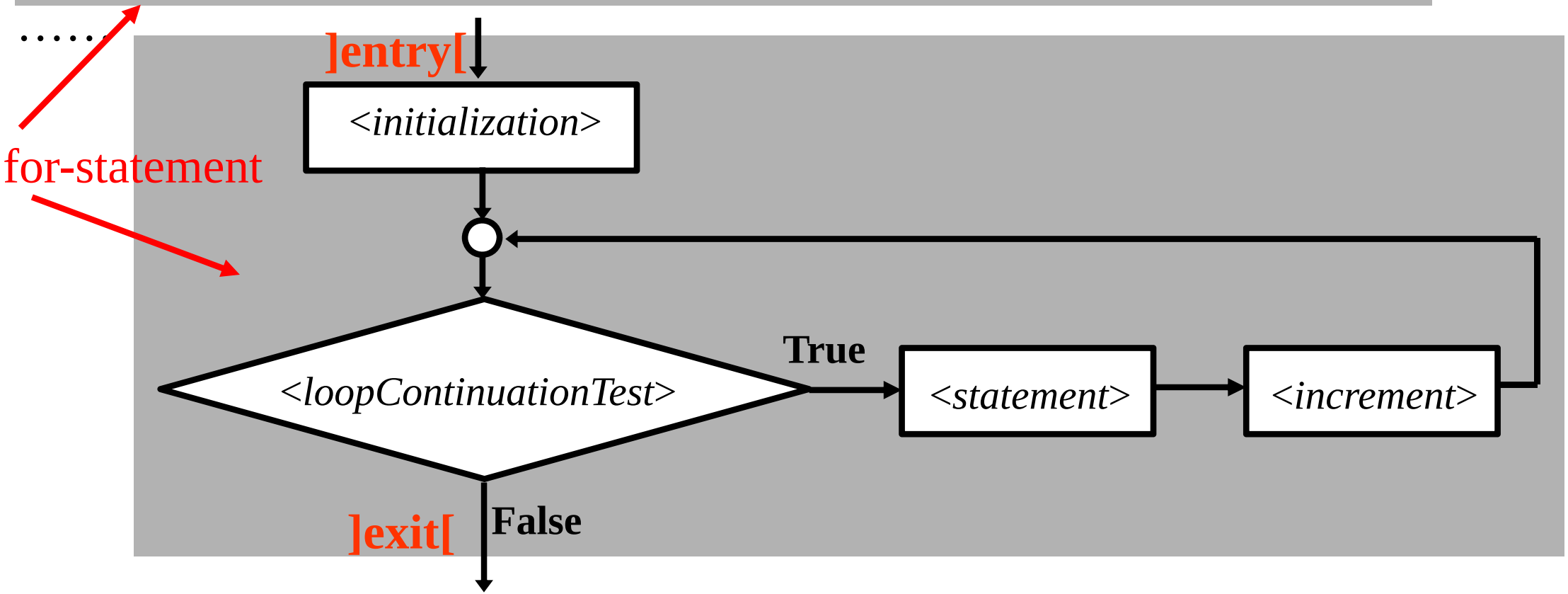
The **do/while** loop with break/continue statement

The for Repetition Structure

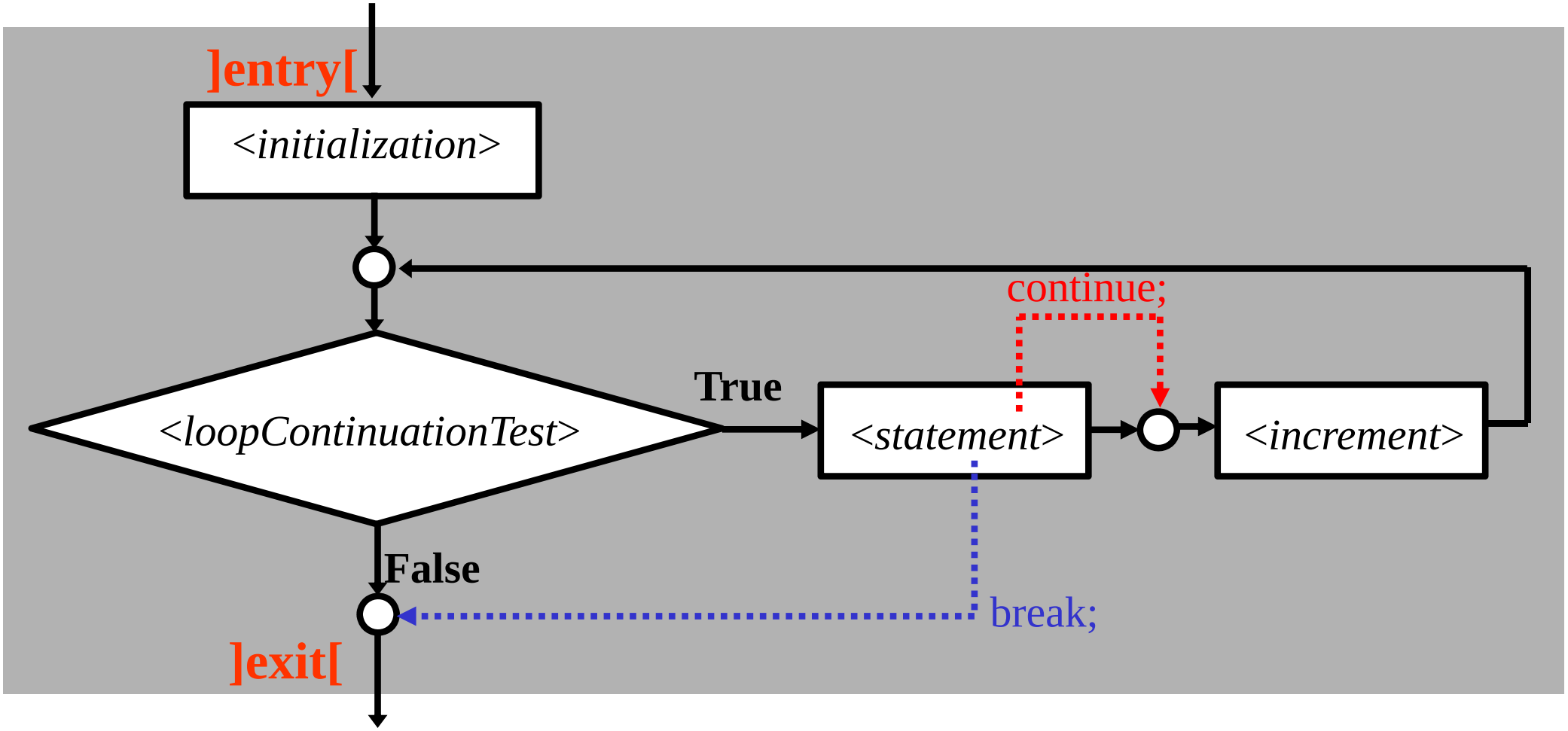
.....

```
for(<initialization>; <loopContinuationTest>; <increment>)
        <statement>
```

.....



The for-break/continue Structure



The **for** loop with break/continue statement

陣列(Arrays) (1/2)

- A set of related data items, called elements, with the same type
- Syntax for declaring arrays

  <type> <array_name>[*<array_size>*];

  E.g. int c[12];

  Similar to normal variables, multiple arrays of the same type can be declared in a single declaration statement

  E.g. int b[100], x[27];
- *<array_size>* should be a **constant** value or expression, i.e., determined in the compilation time

陣列(Arrays) (2/2)

- All the elements of an array have the same name
  - In addition to the *array_name*, referencing to a specific element needs to specify its *position_number*(subscript)

    Syntax: *array_name*[*position_number*]
  - The position numbers are between *0* and *array_size-1*
    - E.g. An *n*-element array c consists of the following elements

      c[0], c[1], …, c[n–1]
- Array elements are like normal variables
  - E.g. c[0] = 3; cout << c[0];

字串(Strings)

- 1-D Arrays of characters ending with **the NULL character ('\0' – ASCII 0，字串終止字元)**，defined in **iostream**
  - E.g.

```
char string1[] = "hello"; //implicitly append '\0'
char string2[] = {'h', 'e', 'l', 'l', 'o' , '\0' };
char string3[] = {'h', 'e', 'l', 'l', 'o'}; //Note!! not a string
```

  - Subscripting is the same as for a normal array
    - E.g.

```
string1[0] is 'h'
string1[2] is 'l'
```

Characters vs. Strings

- Character constants
  - A character enclosed with **single quotes**, e.g., **'z'**
  - A 1-byte unsigned integer, e.g., 'z' represents the integer value of z -- 122 in the ASCII
- String constants
  - Enclosed with **double quotes**, e.g., **"I like C++"**
  - Array of characters, ending with the **NULL** character **'\0'**

2-D Arrays (1/2)

- A 2-D array can be viewed as a table consisting of rows and columns
  - Declaration syntax

    <type> <array_name>[<row_size>][<column_size>]
  - E.g. int a[3][4];

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Column subscript

Array name

Row subscript

2-D Arrays (2/2)

- An *M*x*N* *2*-D array can be viewed as a 1-D array with *M* elements, where each element is a 1-D array with *N* elements
  - E.g.

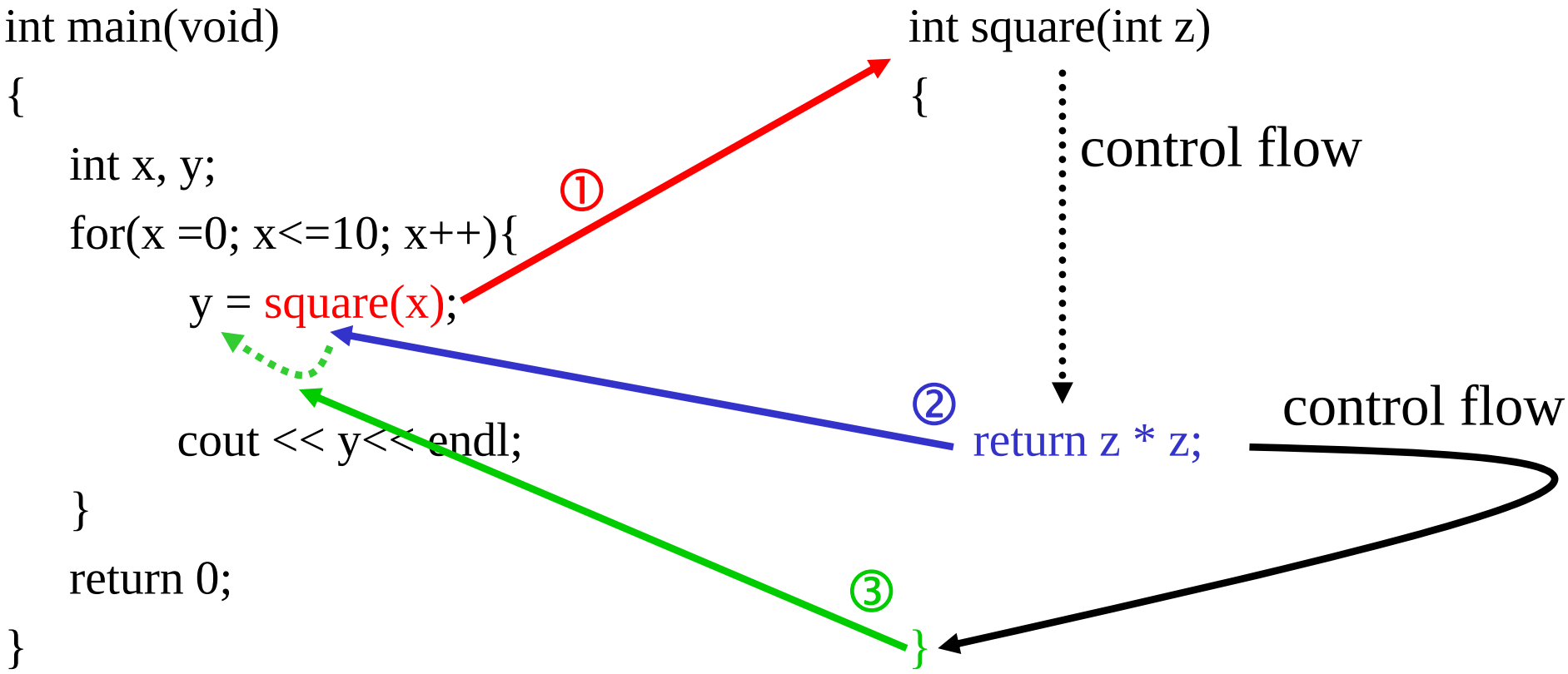
	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

- Array **a** has *3* elements, each element is an array of size *4*
- The order in memory is a[0][0], a[0][1],…, a[2][2], a[2][3] (row-major)

A Function Call

- Functions are invoked by a function call
  - A function call specifies the function name and provides the information (as arguments or parameters) that the called function needs to finish its task
    - E.g. sqrt(a) where "sqrt" is the function name and "a" is the argument needed by function sqrt

Function Operation Model



區域變數(Local Variables)

- All variables declared within functions are local variables
  - The **scope** of local variables is limited to the function in which they are declared
    - Hidden outside the resident function
- A function's arguments are also local variables
  - Get outside information

全域變數(Global Variables)

- Global(or External) variables are declared outside any function definition
  - Retain their values throughout the execution of the program

遞迴(Recursive Functions)

- Functions which may call themselves directly or indirectly (through other functions) are called *recursive functions*
  - E.g.

```
int factorial(int n)    //n! = n * (n-1)!, where 1!=1
{
    int product;
    if(n != 1){
      product = n * factorial(n-1);
    }else{
      product = 1;
    }
    return product;
}
```

結構 (Structures)

- Structures are *aggregated* data-types built using elements of other data-types and/or structures
  - Syntax of the structure definition

```
struct <tag>{
        <elements of other types> //data members
};
```

  - ➢ *<tag>* can be used to declare structure variables of type *<tag>*
  - ➢ *struct <tag>* also can be used to declare structure variables of type *<tag>*

運算子優先權與結合律

Precedence	Operators							Associativity	Type
1	[	]	->	.				left to right	array index
2	++	--	static_cast<type>( )					left to right	unary (postfix)
3	!	+	-	++	--	&	*	right to left	unary (prefix)
4	*	/	%					left to right	multiplicative
5	+	-						left to right	additive
6	<<	>>						left to right	insertion/extraction
7	<	<=	>=	>				left to right	relational
8	==	!=						left to right	equality
9	&&							left to right	logical AND
10	‖							left to right	logical OR
11	?:							right ro left	conditional
12	=	*=	/=	%=	+=	-=		right ro left	assignment
13	,							left to right	comma

結論與期許

- 期望透過舉辦具公信力之「大學程式設計先修檢測」，檢驗具備程式設計能力之高中職學生的學習成果，提供大學作為選才的參考依據。
- 藉由本檢測之推動，除了讓高中職重視資訊科學課程的學習外，亦讓大學酌訂抵免程式設計學分的相關措施。

New! APCS 最新消息

- 下次檢測日期 2020年7月4日,
  報名將自4月27日起...

~謝謝聆聽~

饒建奇