Jiangxi University of Science and Technology

Chapter 4 Selection

Lecture0401 : Relational Expressions

Objectives

THE C PROGRAMMING LANGUAGE

What we will study in chapter 4

4.1 Relational Expressions 关系表达式

4.2 The if and if-else Statements

4.2 The if-else Chain

4.3 The switch Statement

4.4 Case Study: Data Validation

4.7 Chapter Summary

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢ Flow of control 控制流

– ***Flow of control*** refers to ***the order*** in which a program's statements are executed

– ***four standardized flow of control structures***:

1. ***Normal flow of control*** for all programs is **Sequential** 顺序结构
2. **Selection** 选择结构 is used to ***select*** which statements are performed next based on ***a condition***
3. **Repetition** 循环结构 is used to ***repeat*** a set of statements
4. **Invocation** 调用结构 is used to ***invoke*** a sequence of instructions using a single statement, as in calling a function (调用函数)

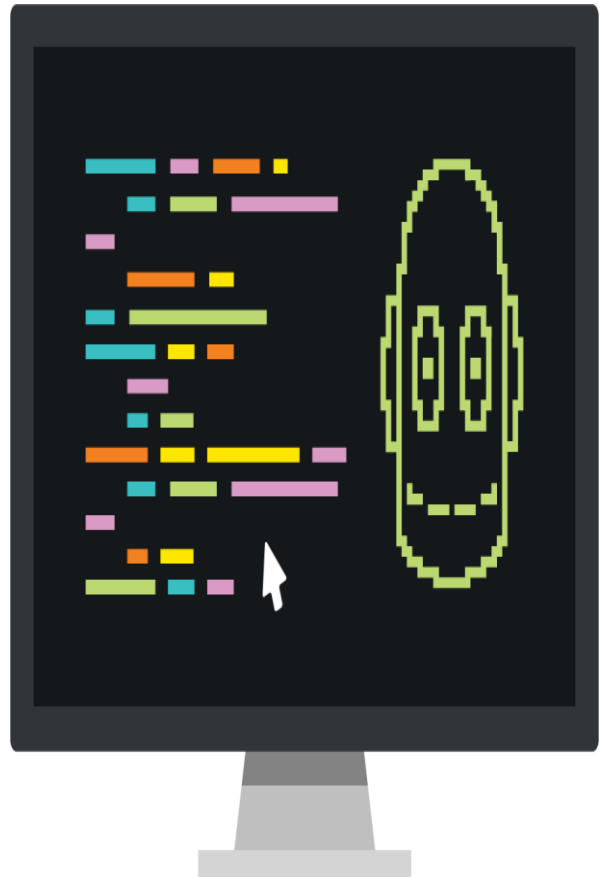
4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢ Simplest decision structure(简单决策结构):

– **if (condition) statement**;

1. statement executed if *condition* 条件 is **true**
2. The condition is evaluated to determine its numerical value, which is interpreted as either **true** (**non-zero**) or **false** (**0**)
3. If condition is true the statement following the if is executed; otherwise, statement is not executed



4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

- **Relational Operator 关系运算符**

Table 4.1 Relational Operators in C

Relational Operator	Meaning	Example
<	less than	age < 30
>	greater than	height > 6.2
<=	less than or equal to	taxable <= 20000
>=	greater than or equal to	temp >= 98.6
==	equal to	grade == 100
!=	not equal to	number != 250

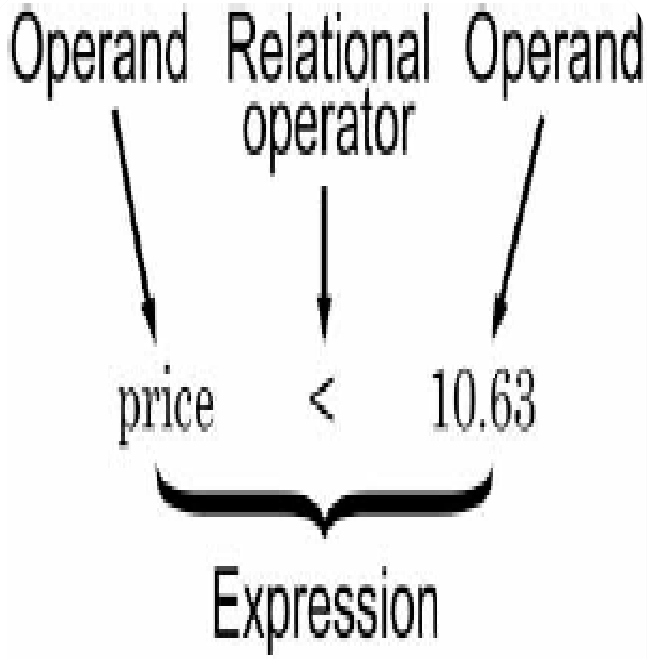


Figure 4.1 Anatomy of a simple relational expression

4.1 Relational Expressions

➢Conditions

- Relational expressions are also known as conditions
- A relational expression evaluates to 1 (**true**) or 0 (**false**)
- The expression **3 < 4** has a value of 1
- The expression **2.0 > 3.3** has a value of 0
- The value of ***hours*** **> 0** depends on the value of ***hours***

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢Conditions

– Character data can also be compared using relational operators

Table 4.2 Sample Comparisons of ASCII Characters

Expression	Value	Interpretation
'A' > 'C'	0	false
'D' <= 'Z'	1	true
'E' == 'F'	0	false
'g' >= 'm'	0	false
'b' != 'c'	1	true
'a' == 'A'	0	false
'B' < 'a'	1	true
'b' > 'Z'	1	true

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢Logical Operators (逻辑运算符)

– More complex conditions can be created using the logical operations

AND (**&&**), **OR** (**||**), and **NOT** (**!**)

– When the **&&** is used with two expressions, the condition is true only if both expressions are true by themselves

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢ Logical Operators (逻辑运算符)

- **int i = 15, j = 30;**
- **double a = 12.0, b = 2.0, complete = 0.0;**

a	b	! a	! b	a && b	a || b
0	0	1	1	0	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	1	1

Watch 1

Name	Value	Type
a>b	true	bool
i==j||a<b||complete	false	bool
a/b>5&&i<20	true	bool

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

- **short-circuit evaluation** 短路求值
  - The evaluation feature for the && or || operators that makes the evaluation of an expression ***stop*** as soon as it is determined that an expression is ***false*** or ***true*** is known as short-circuit evaluation

(6 * 3==36/2) &&(13<3*3+4)||!(6 -2<5)

= (18 == 18) && (13 < 9 + 4) || !(4 < 5)

= 1 && (13 < 13) || !1

= 1 && 0 && 0

= 1 && 0

= 0

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢Precedence and Associativity

Table 4.6 C Operators Listed from Highest Precedence to Lowest Precedence

Operator	Associativity
!, unary -, ++, --	right to left
*, /, %	left to right
+, -	left to right
<, <=, >, >=	left to right
==, !=	left to right
&&	left to right
||	left to right
+=, -=, *=, /=	right to left

4.1 Relational Expressions

THE C PROGRAMMING LANGUAGE

➢Precedence and Associativity

- char key = 'm';
- int i = 5, j = 7, k = 12;
- double x = 22.5;

Expression	Equivalent Expression	Value	Interpretation
i + 2 == k - 1	(i + 2) == (k - 1)	0	false
3 * i - j < 22	((3 * i) - j) < 22	1	true
i + 2 * j > k	(i + (2 * j)) > k	1	true
k + 3 <= -j + 3 * i	(k + 3) <= ((-j) + (3*i))	0	false
'a' + 1 == 'b'	('a' + 1) == 'b'	1	true
key - 1 > 'p'	(key - 1) > 'p'	0	false
key + 1 == 'n'	(key + 1) == 'n'	1	true
25 >= x + 4.0	25 >= (x + 4.0)	0	false

Reference

- BOOK
- Some part of this PPT given by Prof 欧阳城添

(Prof: Chengtian Ouyang)

- with special thank
- https://www.codingunit.com/c-tutorial-first-c-program-hello-world