Branches / Selection

Chapter 4

Logical Expressions

Result is boolean-valued: true or false

- == equal to
- != not equal to
- > greater than
- < less than
- >= greater than or equal to
- <= less than or equal to
- && logical and
- || logical or
- ! logical not

E1 | E2 | E1 & E2
f | f | f
f | t | f
t | t | t
t | t | t

Flow of Control Statements

Three kinds

- Sequential
- Branches (also known as decisions)
  - if
  - if-else
  - switch
- Loops (also known as iterative)
  - for
  - while
  - do-while

Sequential Statements

All statements are executed once, one after another

entry

statement 1

statement 2

statement 3

statement n

exit

if-Statement

Executed if true

if (expression)

statement;

if (line33 < line34)

line37 = line36 - line35;

if (income >= 1400 || interest > 750)

{ must_fall++; deductions = 1;
  adjusted_income = line37;
}

Short Circuit Evaluation

Toward Compact and Efficient Code

- E1 and E2 are boolean-valued expressions
- if (E1 && E2)
  - if E1 is false, the whole expression is false and E2 is not evaluated
- if (E1 || E2)
  - if E1 is true, the whole expression is true and E2 is not evaluated

if (n != 0 && 100 / n > min) <- if (n != 0) if (100 / n > min)
**if-else Statement**

Choose One Statement

- if (expression) statement-1;
- else statement-2;

- if (line18 > line19)
  tax_owed = line18;
- else
  refund = line19;

- if (lineFlag)
  print("%&d",lines);

**True and False**

An Existential Dilemma?

- Any expression may be interpreted as a boolean value
  - 0 is false
  - Non-0 is true
- The result of a “boolean” expression is either 1 or 0
- C++ defines type bool with possible values: true & false
  - Syntactic window-dressing for an int
  - true = 1
  - false = 0

- if (n % 2)
  printf("n is odd\n");
- else
  printf("n is even\n");

- if (strcmp(s1, s2))
  printf("s1 & s2 differ\n");
- else
  printf("s1 & s2 are equal\n");

**Nested Conditional Statement**

Unlimited Nesting Depth

- if (expression-1)
  if (expression-2)
    statement-1;
  else
    statement-2;
  else
    if (expression-3)
      statement-3;
    else
      statement-4;

**The Dangling else Problem**

else attaches to the last if

- if (expression-1)
  if (expression-2)
    statement-1;
  else
    statement-2;

- if (expression-1)
  if expression-2
    statement-1;
  else
    statement-2;

**if-else-if Ladder**

Selects and Executes One Statement

- if (expression-1)
  statement-1;
- else if (expression-2)
  statement-2;
- else if (expression-3)
  statement-3;
- ...
- else if (expression-m)
  statement-m;
- else
  statement-n

**Ctype “Library”**

Character Identification and Mapping

- Using ctype macros
  - Macros return 1 for true and 0 for false
  - isalpha() is defined on all integer values; the rest are defined only for integers representing characters, or EOF
  - #include <ctype.h>

  - int isalpha(int c)
  - int isupper(int c)
  - int islower(int c)
  - int isxdigit(int c)
  - int isalnum(int c)
  - int isalnum(int c)
  - int tolower(int c)

  - int ispunct(int c)
  - int isprint(int c)
  - int isgraph(int c)
  - int iscntrl(int c)
  - int isascii(int c)
  - int toupper(int c) /* function */
  - int isspace(int c) /* function */
**Conditional Expressions**

Based on the Conditional Operator ?:

- `(expr 1) ? (expr 2) : (expr 3)`
  - if `expr 1` is true, `expr 2` is the value of the overall expression
  - if `expr 1` is false, `expr 3` is the value of the overall expression
  - Parentheses are not syntactically required
  - Typically used because `?` has a high precedence

- `max = (x > y) ? x : y;`
- `min = (x < y) ? x : y;`
- `index = (index+1 == size) ? 0 : ++index;`

---

**switch Example**

Excerpt from `wc.c`

```
/* counts characters, words, and lines in a file */

switch(c)
  { case 'n': lines++; /* fall through */
    case ' ': inword = 0;
    case 't':
      inword = 0;
      break;
    default:
      if (!inword) /* start word */
        { inword = 1;
          words++;
        }
      break;
  }
```