Control Flow

Devendra Tewari

2024-10-09

if-else

- `if-else` expresses decisions
- `else` part is optional

```
if (expression)
statement1
else
statement2
```

- `expression` returns a numerical value
- 0 is considered FALSE, any other value is TRUE

if-else ambiguities

- By default, the `else` is associated with the inner `if`

```
if (i >= 0)
  if (i < 5)
    a = b;
  else
    a = c;
```

- Use braces to remove ambiguity

```
if (i >= 0) {
  if (i < 5)
    a = b;
} else
  a = c;
```

else-if

- Useful for expressing multi-way decisions

```
if (expression)
  statement
else if (expression)
  statement
else if (expression)
  statement
else
  statement
```

switch

```
switch (expression) {
  case const-expr:
    statements
  case const-expr:
    statements
  default:
    statements
}
```

- Used to express multi-way decision
- Matches the result of an expression to one of several integer constants
- a `break` statement causes exit from the `switch`
- without a `break` all statements after the matching `case` are executed till the end of the `switch` block

while

- The while loop executes as long as `expression` is TRUE (not 0)

```
while (expression) {
  statements
}
```

for

- The loop has three parts

  ```
  for (expr1; expr2; expr3) {
    statements
  }
  ```

  - `expr1` is an initialization expression
  - `expr2` is a relational expression, and
  - `expr3` is the increment expression
- The loop executes as long as `expr2` is TRUE (not 0)
- All three expressions can be empty which leads to an infinite `for` loop

do-while

- The `do` loop executes at least once before `expression` is evaluated

```
do {
  statements
} while (expression);
```

- The loop executes as long as `expression` is TRUE (not 0)

break

- Using the break statement causes immediate exit from a loop (`for`, `while` or `do-while`) or `switch` block

continue

- The `continue` statement causes a loop to begin the next iteration, the statements following `continue` are not executed

goto

- the `goto` statement causes execution to jump to the statements after the `label`

  ```
  goto label;
  statements
  label:
  statements
  ```

- `goto` is not recommended as it results in spaghetti code

Exercise

- Write a program that converts 1 to 50 mile(s) into kilometers

  NOTE 1 mile = 1.609344 kilometers

- Print the result in tabular form

```
01 mile(s) = 01.609344 km   02 mile(s) = 03.218688 km
03 mile(s) = 04.828032 km   04 mile(s) = 06.437376 km
```