Trần Trung Hiếu – Báo cáo thực tập

26 – 30/12/2022

**Flow Control**

Trình biên dịch Java thực thi mã từ trên xuống dưới. Các câu lệnh trong mã được thực hiện theo thứ tự mà chúng xuất hiện. Tuy nhiên, Java cung cấp các câu lệnh có thể được sử dụng để kiểm soát dòng lệnh. Các câu lệnh như vậy được gọi là các câu lệnh luồng điều khiển. Đây là một trong những tính năng cơ bản của Java.

Java cung cấp ba loại câu lệnh luồng điều khiển.

* Decision Making statements
  + if statements
  + switch statement
* Loop statements
  + do while loop
  + while loop
  + for loop
  + for-each loop
* Jump statements
  + break statement
  + continue statement

1. If statements

Trong Java, câu lệnh "if" được sử dụng để đánh giá một điều kiện. Điều kiện của câu lệnh If đưa ra một giá trị Boolean, đúng hoặc sai. Trong Java, có bốn loại câu lệnh if được đưa ra dưới đây:

* if statement

Đây là câu lệnh cơ bản nhất trong số tất cả các câu lệnh luồng điều khiển trong Java. Nó đánh giá một giá trị Boolean và cho phép chương trình nhập một khối mã nếu biểu thức có giá trị là true.

if(condition) {

statement 1;

}

* if-else statement

Câu lệnh if-else là một phần mở rộng của câu lệnh if. Khối (block) else được thực thi nếu điều kiện của khối if được đánh giá là sai.

* Chuỗi if-else-if

Câu lệnh if-else-if chứa câu lệnh if theo sau là nhiều câu lệnh else-if. Nói cách khác, việc này tạo ra cây quyết định (decision tree) cho chương trình thực thi khi có 1 điều kiện trong chuỗi là đúng

* Nested if-statement

Trong các câu lệnh if lồng nhau (nested), câu lệnh if có thể chứa một câu lệnh if hoặc if-else bên trong một câu lệnh if hoặc else-if khác.

if (condition 1) {

statement 1; //executes when condition 1 is true

if (condition 2) {

statement 2; //executes when condition 2 is true

} else {

statement 2; //executes when condition 2 is false

}

}

1. Switch

Trong Java, câu lệnh Switch tương tự như câu lệnh if-else-if. Câu lệnh switch chứa nhiều khối mã được gọi là các case và một case duy nhất được thực thi dựa trên biến đang được switch. Câu lệnh switch dễ sử dụng hơn và dễ đọc hơn thay vì câu lệnh if-else-if.

Các điểm cần lưu ý về câu lệnh switch:

* Các biến của case có thể là int, short, byte, char hoặc enum. Kiểu chuỗi cũng được hỗ trợ kể từ phiên bản 7 của Java.
* Các case không thể trùng lặp
* Câu lệnh mặc định được thực thi khi không case nào khớp với giá trị của biểu thức. Câu lệnh mặc định là tùy chọn.
* Câu lệnh break kết thúc khối switch khi điều kiện được thỏa mãn. Break cũng là tùy chọn, và nếu không được sử dụng, case tiếp theo sẽ được thực thi.

switch (expression){

case value1:

statement1;

break;

.

case valueN:

statementN;

break;

default:

default statement;

}

1. Loop

* for loop

Trong Java, vòng lặp for tương tự như trong C và C++. Nó cho phép chúng ta khởi tạo biến vòng lặp, kiểm tra điều kiện và tăng/giảm trong một dòng mã. Chỉ sử dụng vòng lặp for khi biết chính xác số lần ta muốn thực thi khối mã.

for(initialization, condition, increment/decrement) {

//block of statements

}

* for-each loop

Java cung cấp một vòng lặp for nâng cao để duyệt qua các cấu trúc dữ liệu như mảng hoặc tập hợp. Trong vòng lặp for-each, chúng ta không cần cập nhật biến vòng lặp.

for(data\_type var : array\_name/collection\_name){

//statements

}

* while loop

Vòng lặp while cũng được sử dụng để lặp lại số lượng câu lệnh nhiều lần. Tuy nhiên, nếu chúng ta không biết trước số lần lặp thì nên sử dụng vòng lặp while. Không giống như vòng lặp for, việc khởi tạo và tăng/giảm không diễn ra bên trong câu lệnh vòng lặp while.

Nó còn được gọi là vòng lặp kiểm soát đầu (entry-controlled loop) vì điều kiện được kiểm tra khi bắt đầu vòng lặp. Nếu điều kiện đúng thì thân vòng lặp sẽ được thực hiện.

while(condition){

//looping statements

}

* do-while loop

Vòng lặp do-while kiểm tra điều kiện ở cuối vòng lặp sau khi thực hiện các câu lệnh của vòng lặp. Khi không biết số lần lặp và chúng ta phải thực hiện vòng lặp ít nhất một lần, chúng ta có thể sử dụng vòng lặp do-while.

Nó còn được gọi là vòng lặp kiểm soát cuối (exit-controlled loop) vì điều kiện không được kiểm tra trước.

1. Break

Lệnh break được sử dụng để ngắt luồng hiện tại của chương trình và chuyển điều khiển sang câu lệnh tiếp theo bên ngoài câu lệnh loop hoặc switch. Tuy nhiên, nó chỉ phá vỡ vòng lặp bên trong trong trường hợp của nested loop.

Câu lệnh break không thể được sử dụng độc lập trong chương trình Java, nghĩa là, nó chỉ có thể được viết bên trong câu lệnh vòng lặp hoặc chuyển đổi.

public class BreakExample {

public static void main(String[] args) {

for (int i = 0; i <= 10; i++){

System.out.println(i);

if(i==6) {

break;

}

}

}

}

1. Continue

Không giống như câu lệnh break, câu lệnh continue không phá vỡ vòng lặp, ngược lại, nó bỏ qua phần cụ thể của vòng lặp và chuyển ngay đến lần lặp tiếp theo của vòng lặp.

for(int i = 0; i<= 2; i++) {

for (int j = i; j<=5; j++) {

if(j == 4) {

continue;

}

System.out.print(j + “ “);

}

}

////Output

0 1 2 3 5 1 2 3 5 2 3 5