Control Statements - Repetition/Looping Structures

Student's Name
University
Course
Professor
Date
Control Statements - Repetition/Looping Structures

Introduction

Usually, statements in a program are executed line by line in the sequence of their occurrences. This is known as *sequential execution*, as shown in Fig 1 (Deitel & Deitel, 2017). However, some statements enable us to specify that sequential execution can be avoided as the need arises. This is known as the *transfer of control*, as shown in Fig 2 (Deitel & Deitel, 2017).

Figure 1: Sequence-structure activity diagram

![Sequence-structure activity diagram](image1)

Figure 2: while repetition statement UML activity diagram

![while repetition statement UML activity diagram](image2)

In a loop, a part of the program is repeated over and over until a specific goal is achieved. Loops are significant for calculations that require repeated procedures for processing large
data (Horstmann, 2013). For instance, printing odd/even numbers in a list of numbers between 0 and 20 inclusive. Loop implementation differs in every programming language. Java programming language, for example, has three different implementations, that is, do-while, while, and for (Deitel & Deitel, 2017). Others, like python, support only for and while loops.

**Types of Loops**

1. **do-while loop**

   The do-while loop executes at least once, and it continues the execution until a given condition is achieved. It is also called a post-tested loop. It is mostly used with menu-driven programs. The do-while loop syntax for a Java program is:

   ```java
   do {
     // code to execute
   } while(condition);
   ```

2. **while loop**

   Also referred to as a pre-tested loop. It is used in cases where the expected number of iterations is not known from the start. The code block is executed until the condition specified in the loop is satisfied. The while loop syntax for a Java program is:

   ```java
   while (condition) {
     // code to execute
   }
   ```

3. **for loop**

   It allows for the execution of some parts of the program until the given condition is fulfilled. It is mostly used for a known number of iterations. The for-loop syntax for a Java program is:

   ```java
   for (initialization; condition; increment/decrement) {
   ```
Advantages of loops

1. Provides code reusability.
2. Allows us to avoid code duplication.
3. Loops allow for traversing or iterating over elements of data structures, for instance, arrays or linked lists.

Disadvantages

1. The increased time complexity of an algorithm, the Big O notation.

Python Code Sample

```python
# code to execute
}

# create a list of integers 0-20 inclusive
nums = list(range(0, 21))

# create a list of odd numbers
odds = []
for i in nums:
    if i % 2 == 1:
        odds.append(i)

# iterate through a list using for loop
if i % 2 == 0:
    evens.append(i)

# create a list of odd numbers
else:
    odds.append(i)

# print numbers
print('List of Even Numbers')
print(evens)
print('List of Odd Numbers')
print(odds)
```

Figure 3: Python code sample to print odd and even numbers between 0 and 20
Figure 3 shows a python code screenshot that iterates through an array (list) and creates other lists of even and odd numbers. Figure 4 is the output of running the python code in Figure 3.