115-[PF] - 실습 - 루프

실습 개요

루프는 코드에서 반복되는 세그먼트입니다. 2 가지 유형의 루프인 `while` 루프와 `for` 루프를 소개합니다.

본 실습에서는 다음을 수행합니다.

- `while` 루프 사용
- `for` 루프 사용

예상 완료 시간

45 분

AWS Cloud9 IDE 액세스

1. 이 지침의 상단으로 이동한 다음 **Start Lab** 을 선택하여 실습 환경을 시작합니다.

   Start Lab 패널이 열리고 실습 상태가 표시됩니다.

2. *Lab status: ready* 라는 메시지가 표시되면 **X** 를 선택하여 **Start Lab** 패널을 닫습니다.
3. 지침의 맨 위에서 **AWS** 를 선택합니다.

   새 브라우저 탭에서 AWS 관리 콘솔이 열립니다. 시스템에 자동으로 로그인됩니다.

참고: 새 브라우저 탭이 열리지 않는 경우 일반적으로 브라우저에서 팝업 창을 열 수 없음을 나타내는 배너 또는 아이콘이 브라우저 상단에 표시됩니다. 배너 또는 아이콘을 선택하고 **Allow pop ups** 를 선택합니다.

4. AWS 관리 콘솔에서 **Services** > **Cloud9** 을 선택합니다. **Your environments** 패널에서 **reStart-python-cloud9** 카드를 찾아 **Open IDE** 를 선택합니다.

   AWS Cloud9 환경이 열립니다.

   참고: *.c9/project.settings have been changed on disk* 라는 메시지가 담긴 팝업 창이 표시되면 **Discard** 를 선택하여 무시합니다. 마찬가지로, *Show third-party content* 라는 대화 창이 나타나면 **No** 를 선택하여 거절합니다.

Python 연습 파일 생성

5. 메뉴 모음에서 **File > New From Template > Python File** 을 선택합니다.

   이 작업은 제목이 없는 파일을 생성합니다.

6. 템플릿 파일에서 샘플 코드를 삭제합니다.
7. **File > Save As...**를 선택하고, 연습 파일에 적절한 이름(예: *while-loop.py*)을 입력한 다음 **/home/ec2-user/environment** 디렉터리에 저장합니다.

터미널 세션에 액세스

8. AWS Cloud9 IDE 에서 **+** 아이콘을 선택하고 **New Terminal** 을 선택합니다.

   터미널 세션이 열립니다.

9. 현재 작동 중인 디렉터리를 표시하려면 `pwd` 를 입력합니다. 이 명령은 **/home/ec2-user/environment** 를 가리킵니다.
10. 이 디렉터리에서 이전 섹션에서 생성한 파일을 찾을 수 있어야 합니다.

연습 1: while 루프 작업

`while` 루프는 특정 조건을 충족할 때까지 코드의 섹션이 반복되도록 합니다. 이 연습에서는 사용자에게 올바르게 숫자를 추측하라고 요청하는 Python 스크립트를 생성합니다.

게임 규칙 출력

11. IDE 의 탐색 창에서 이전 *Python 연습 파일 생성* 섹션에서 생성한 **.py** 파일을 선택합니다.
12. `print()` 함수를 사용하여 사용자에게 게임에 대해 알립니다.

```
print("Welcome to Guess the Number!")
print("The rules are simple. I will think of a number, and you will try to guess it.")
```

13. 파일을 저장하고 실행합니다.
14. 스크립트가 올바르게 실행되고 출력이 예상한 대로 표시되는지 확인합니다.

random 가져오기 및 while 루프 작성

`import` 명령을 사용하여 다른 사람이 작성한 코드를 포함하도록 합니다. 지금까지는 내장된 함수를 사용했습니다. 함수는 재사용할 수 있는 코드 조각이라는 것을 기억하십시오.

15. 파일 상단에서 `random`(이)라는 Python 모듈(라이브러리의 한 유형임)을 가져옵니다.

    참고: 일반적으로 `import` 스테이트먼트는 스크립트 상단에 위치합니다.

```
import random
```

16. 커서를 2 번째 `print()` 스테이트먼트의 다음 줄에 놓습니다. 그런 다음, `random` 모듈의 `randint()` 함수를 사용하여 1~10 사이의 임의 숫자를 생성하는 스테이트먼트를 입력합니다.

```
number = random.randint(1,10)
```

17. *isGuessRight* 라는 변수를 생성하여 사용자가 숫자를 제대로 추측했는지 추적합니다.

```
isGuessRight = False
```

18. 게임 논리를 처리하려면 `while` 루프를 생성합니다.

```
while isGuessRight != True:
    guess = input("Guess a number between 1 and 10: ")
    if int(guess) == number:
        print("You guessed {}. That is correct! You win!".format(guess))
        isGuessRight = True
    else:
        print("You guessed {}. Sorry, that isn't it. Try again.".format(guess))
```

    참고: `while` 루프는 숫자를 올바르게 추측할 때까지 루프 내에서 코드를 반복하며, 이는 코드에서 `isGuessRight != True` 조건으로 나타납니다. 또한, Python 은 들여쓰기를 사용하여 논리 블록을, 혹은 어떤 스테이트먼트가 `while` 루프의 일부로 간주되는지 결정합니다. 커서를 스테이트먼트 옆에 놓고 TAB 키를 눌러 줄을 들여 쓸 수 있습니다.

19. 파일을 저장합니다.

의사코드 작성

Python 스크립트를 실행하기 전에 작성된(비코드) 문장으로 `while` 루프의 논리를 작성합니다. 이 기법을 *의사코드*라고 합니다.

예를 들면 다음과 같습니다.

- 사용자가 답변을 올바르게 추측하지 않았다면 루프로 들어갑니다.
- 사용자에게 추측하라고 요청합니다.
- 추측이 올바른 숫자입니까?
- 추측이 올바르면 사용자에게 추측이 올바르다고 알리고 루프를 나갑니다.
- 추측이 틀리면 사용자에게 추측이 틀렸다고 알리고 루프를 계속합니다.

스크립트 실행

이제 Python 스크립트를 실행하고 작동하는지 확인합니다.

20. 파일을 실행합니다.
21. 스크립트가 올바르게 실행되고 출력이 예상한 대로 표시되는지 확인합니다.

주석 추가

코드에 주석을 작성하는 것이 유용할 수 있습니다. Python 에서 주석 줄은 무시되며, 우물정자 기호(#)로 시작합니다. 대부분의 키보드에서 SHIFT+3 을 눌러 이 기호를 입력할 수 있습니다. 코드에 주석을 추가하면 코드의 기능을 기억하는 데 도움이 됩니다.

사용자에게 스크립트에 대해 알리기

이 활동에서는 사용자에게 스크립트가 수행할 것이 무엇인지 알리는 초기 출력을 생성하여 새 Python 스크립트를 시작합니다.

22. 메뉴 모음에서 **File > New From Template > Python File** 을 선택합니다.
23. 템플릿 파일에 제공된 샘플 코드를 삭제합니다.
24. **File > Save As...**를 선택하고 *for-loop.py* 로 저장합니다.
25. 사용자에게 스크립트에 대해 알리려면 `print()` 함수를 사용합니다.

```
print("Count to 10!")
```

26. 파일을 저장하고 실행합니다.
27. 스크립트가 올바르게 실행되고 출력이 예상한 대로 표시되는지 확인합니다.

for 루프 작성

Python 에서 몇 가지 단어로 많은 기능을 포함할 수 있습니다. 이 기능 때문에 Python 은 다른 프로그래밍 언어에 비해 상대적으로 작성하기 쉽지만, 이로 인해 Python 코드가 읽기 더 어려워질 수 있습니다. 이 활동에서는 `for` 스테이트먼트를 사용하고, 실행한 후에 이를 분석해보는 시간을 갖겠습니다.

28. Python 스크립트로 돌아갑니다. 10 까지 세려면 다음 코드를 입력합니다.

    참고: Python 은 들여쓰기를 사용하여 `print` 스테이트먼트가 `for` 루프 스테이트먼트 안에 있음을 결정합니다.

```
for x in range (0, 11):
    print(x)
```

29. 파일을 저장하고 실행합니다.
30. 스크립트가 올바르게 실행되고 출력이 예상한 대로 표시되는지 확인합니다.

    해당 두 줄에서 무엇이 발생했는지 설명합니다. `for` 스테이트먼트는 `for ... in` 키워드를 사용하여 컴퓨터에 목록을 검토하라고 지시합니다. 목록은 `range()` 함수에 의해 생성됩니다. `range()` 함수는 시작 번호와 종료 번호를 취하지만, 종료 번호는 포함되지 않습니다. 따라서 함수가 10 까지 세고 중지하도록 하려면 `11` 을 전달합니다. *x* 문자는 변수로 기능합니다. 루프를 실행할 때마다 변수 *x* 는 루프의 다음 변수에 할당되고 화면에 출력됩니다.

축하합니다! Python 에서 `while` 및 `for` 루프를 작업했습니다.

실습 종료

축하합니다! 실습을 마치셨습니다.

31. 이 페이지의 상단에서 **End Lab** 을 선택한 다음 Yes 를 선택하여 실습 종료를 확인합니다.

    *DELETE has been initiated... You may close this message box now.*라는 내용의 패널이 표시됩니다.

32. *Ended AWS Lab Successfully* 라는 메시지가 잠시 표시되어 실습이 종료되었음을 나타냅니다.

추가 리소스

AWS Training and Certification 에 대한 자세한 내용은 https://aws.amazon.com/training/을 참조하십시오.

여러분의 피드백을 환영합니다. 제안이나 수정 사항을 공유하려면 AWS Training and Certification Contact Form 에서 세부 정보를 제공해 주십시오.