# 130- [PF] - 실습 - Hello World 및 시저 암호화 디버깅

# 시저 암호 프로그램 디버깅

## 실습 개요

디버거는 다른 프로그램의 버그(디버그)를 테스트하고 찾는데 사용되는 컴퓨터 프로그램이라는 사실을 기억하십시오. 이 실습에서는 Python Debugger(pdb)를 사용하여 Python 프로그램의 버그를 찾아 수정합니다.

본 실습에서는 다음을 수행합니다.

- Python Debugger 사용
- 이전 실습에서 생성한 시저 암호 프로그램의 서로 다른 버전을 디버깅

## 예상 완료 시간

60 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.
6. 템플릿 파일에서 샘플 코드를 삭제합니다.
7. **File > Save As...**를 선택하고, 연습 파일에 적절한 이름(예: *debug-caesar-1.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 찾습니다.

## 연습 1: 버그가 있는 시저 암호 프로그램 작업 - 1 부

Functions 실습에서 시저 암호 프로그램을 생성하여 메시지를 암호화하고 암호화를 해제했습니다. 이 실습에서는 Python Debugger(pdb)를 사용하여 버그가 있는 프로그램 버전을 찾아 오류를 수정합니다.

11. IDE 의 탐색 창에서 이전 *Python 연습 파일 생성* 섹션에서 생성한 **.py** 파일을 선택합니다. 다음 코드를 복사하여 파일에 붙여 넣습니다.

```
# Module Lab: Caesar Cipher Program Bug #1
#
# In a previous lab, you created a Caesar cipher program. This version of
# the program is buggy. Use a debugger to find the bug and fix it.

# Double the given alphabet
def getDoubleAlphabet(alphabet):
    doubleAlphabet = alphabet + alphabet
    return doubleAlphabet

# Get a message to encrypt
def getMessage():
    stringToEncrypt = input("Please enter a message to encrypt: ")
    return stringToEncrypt

# Get a cipher key
def getCipherKey():
    shiftAmount = input("Please enter a key (whole number from 1-25): ")
    return shiftAmount

# Encrypt message
def encryptMessage(message, cipherKey, alphabet):
    encryptedMessage = ""
    uppercaseMessage = ""
    uppercaseMessage = message.upper()
    for currentCharacter in uppercaseMessage:
        position = alphabet.find(currentCharacter)
        newPosition = position + cipherKey
        if currentCharacter in alphabet:
            encryptedMessage = encryptedMessage + alphabet[newPosition]
        else:
            encryptedMessage = encryptedMessage + currentCharacter
    return encryptedMessage

# Decrypt message
def decryptMessage(message, cipherKey, alphabet):
```

```
    decryptKey = -1 * int(cipherKey)
    return encryptMessage(message, decryptKey, alphabet)

# Main program logic
def runCaesarCipherProgram():
    myAlphabet="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    print(f'Alphabet: {myAlphabet}')
    myAlphabet2 = getDoubleAlphabet(myAlphabet)
    print(f'Alphabet2: {myAlphabet2}')
    myMessage = getMessage()
    print(myMessage)
    myCipherKey = getCipherKey()
    print(myCipherKey)
    myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
    print(f'Encrypted Message: {myEncryptedMessage}')
    myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey,
myAlphabet2)
    print(f'Decrypted Message: {myDecryptedMessage}')

# Main logic
runCaesarCipherProgram()
```

12. 파일을 저장합니다.
13. 버그가 있는 첫 번째 시저 암호 프로그램 실행을 시도합니다. 다음 예시와 비슷한 오류가 표시됩니다.

```
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: AWS Restart rocks
AWS Restart rocks
Please enter a key (whole number from 1-25): 2
2
Traceback (most recent call last):
  File "/home/ec2-user/environment/caesar_cipher_program_bug_1.py", line 56, in
<module>
    runCaesarCipherProgram()
```

```
  File "/home/ec2-user/environment/caesar_cipher_program_bug_1.py", line 50, in
runCaesarCipherProgram
    myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
  File "/home/ec2-user/environment/caesar_cipher_program_bug_1.py", line 28, in
encryptMessage
    newPosition = position + cipherKey
TypeError: unsupported operand type(s) for +: 'int' and 'str'

Process exited with code: 0
```

프로그램은 *역추적*으로 끝납니다. 역추적은 예외 핸들러의 지점에서 시작하는 스택 추적입니다. 그런 다음 호출 체인의 아래로 예외가 발생한 지점까지 내려갑니다. 즉, 오류가 발생한 지점까지 내려갑니다.

14. 디버거를 사용하여 버그가 있는 시저 암호에 대한 첫 번째 실습 파일의 버그를 찾아 수정합니다.

## 연습 2: 버그가 있는 시저 암호 프로그램 작업 - 2 부

역추적은 줄 수와 같은 유용한 단서를 제공하기 때문에 역추적을 야기하는 오류는 일반적으로 수정하기 더 쉽습니다.

15. 메뉴 모음에서 **File > New From Template > Python File** 을 선택합니다.
16. 템플릿 파일에서 샘플 코드를 삭제합니다.
17. **File > Save As...**를 선택하고, 연습 파일에 적절한 이름(예: *debug-caesar-2.py*)을 입력한 다음 **/home/ec2-user/environment** 디렉터리에 저장합니다.
18. 다음 코드를 복사하여 새로 생성한 Python 파일에 붙여 넣습니다.

```
# Module Lab: Caesar Cipher Program Bug #2
#
# In a previous lab, you created a Caesar cipher program. This version of
# the program is buggy. Use a debugger to find the bug and fix it.

# Double the given alphabet
def getDoubleAlphabet(alphabet):
    doubleAlphabet = alphabet + alphabet
```

```
    return doubleAlphabet

# Get a message to encrypt
def getMessage():
    stringToEncrypt = input("Please enter a message to encrypt: ")
    return stringToEncrypt

# Get a cipher key
def getCipherKey():
    shiftAmount = input("Please enter a key (whole number from 1-25): ")
    return shiftAmount

# Encrypt message
def encryptMessage(message, cipherKey, alphabet):
    encryptedMessage = ""
    uppercaseMessage = ""
    uppercaseMessage = message
    for currentCharacter in uppercaseMessage:
        position = alphabet.find(currentCharacter)
        newPosition = position + int(cipherKey)
        if currentCharacter in alphabet:
            encryptedMessage = encryptedMessage + alphabet[newPosition]
        else:
            encryptedMessage = encryptedMessage + currentCharacter
    return encryptedMessage

# Decrypt message
def decryptMessage(message, cipherKey, alphabet):
    decryptKey = -1 * int(cipherKey)
    return encryptMessage(message, decryptKey, alphabet)

# Main program logic
def runCaesarCipherProgram():
    myAlphabet="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    print(f'Alphabet: {myAlphabet}')
    myAlphabet2 = getDoubleAlphabet(myAlphabet)
    print(f'Alphabet2: {myAlphabet2}')
```

```
    myMessage = getMessage()
    print(myMessage)
    myCipherKey = getCipherKey()
    print(myCipherKey)
    myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
    print(f'Encrypted Message: {myEncryptedMessage}')
    myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey,
myAlphabet2)
    print(f'Decrypted Message: {myDecryptedMessage}')

# Main logic
runCaesarCipherProgram()
```

19. 파일을 저장합니다.
20. 버그가 있는 두 번째 시저 암호 프로그램을 실행합니다. 프로그램이 올바르게 종료되는 것처럼 보이지만, 출력을 다시 확인합니다. 예시에서처럼 메시지가 일부만 암호화되었습니다.

```
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: AWS Restart rocks!
AWS Restart rocks!
Please enter a key (whole number from 1-25): 2
2
Encrypted Message: CYU Testart rocks!
Decrypted Messgae: AWS Restart rocks!

Process exited with code: 0
```

21. 디버거를 사용하여 프로그램을 살펴보고 버그를 찾아 봅니다.
22. 버그에 대한 힌트를 얻으려면 여러 가지 입력으로 프로그램을 여러 번 실행합니다. 무엇을 발견했습니까?
23. 프로그램을 실행하고 여러 입력을 입력하여 버그를 찾아 수정하고 수정 내용을 검증합니다.

# 연습 3: 버그가 있는 시저 암호 프로그램 작업 - 3 부

이 연습에서는 시저 암호 프로그램의 버그가 있는 세 번째 버전을 디버그합니다.

24. 메뉴 모음에서 **File > New From Template > Python File** 을 선택합니다.
25. 템플릿 파일에서 샘플 코드를 삭제합니다.
26. **File > Save As...**를 선택하고, 연습 파일에 적절한 이름(예: *caesar_debug-3.py*)을 입력한 다음 **/home/ec2-user/environment** 디렉터리에 저장합니다.
27. 다음 코드를 복사하여 새로 생성한 Python 파일에 붙여 넣습니다.

```
# Module Lab: Caesar Cipher Program Bug #3
#
# In a previous lab, you created a Caesar cipher program. This version of
# the program is buggy. Use a debugger to find the bug and fix it.

# Double the given alphabet
def getDoubleAlphabet(alphabet):
    doubleAlphabet = alphabet + alphabet
    return doubleAlphabet

# Get a message to encrypt
def getMessage():
    stringToEncrypt = input("Please enter a message to encrypt: ")
    return stringToEncrypt

# Get a cipher key
def getCipherKey():
    shiftAmount = input("Please enter a key (whole number from 1-25): ")
    return shiftAmount

# Encrypt message
def encryptMessage(message, cipherKey, alphabet):
    encryptedMessage = ""
    uppercaseMessage = ""
    uppercaseMessage = message.upper()
    for currentCharacter in uppercaseMessage:
        position = alphabet.find(currentCharacter)
```

```
            newPosition = position + int(cipherKey)
            if currentCharacter in alphabet:
                encryptedMessage = encryptedMessage + alphabet[newPosition]
            else:
                encryptedMessage = encryptedMessage + currentCharacter
        return encryptedMessage

# Decrypt message
def decryptMessage(message, cipherKey, alphabet):
        decryptKey = -1 * int(cipherKey)
        return encryptMessage(message, cipherKey, alphabet)

# Main program logic
def runCaesarCipherProgram():
        myAlphabet="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
        print(f'Alphabet: {myAlphabet}')
        myAlphabet2 = getDoubleAlphabet(myAlphabet)
        print(f'Alphabet2: {myAlphabet2}')
        myMessage = getMessage()
        print(myMessage)
        myCipherKey = getCipherKey()
        print(myCipherKey)
        myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
        print(f'Encrypted Message: {myEncryptedMessage}')
        myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey,
myAlphabet2)
        print(f'Decrypted Message: {myDecryptedMessage}')

# Main logic
runCaesarCipherProgram()
```

28. 파일을 저장합니다.
29. 버그가 있는 세 번째 시저 암호 프로그램 실행을 시도합니다. 출력은 거의 올바른 것처럼 보입니다. 그러나 다음 예에서 볼 수 있듯이 *AWS Restart* 메시지의 비암호화는 올바르지 않습니다.

```
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

```
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: AWS Restart rocks!
AWS Restart rocks!
Please enter a key (whole number from 1-25): 2
2
Encrypted Message: CYU TGUVCTV TQEMU!
Decrypted Message: EAW VIWXEVX VSGOW!

Process exited with code: 0
```

30. 디버거를 다시 시작할 시간입니다! 버그를 찾아 수정합니다.

## 연습 4: 버그가 있는 시저 암호 프로그램 작업 - 4 부

이 연습에서는 시저 암호 프로그램의 버그가 있는 네 번째이자 마지막 버전을 디버그합니다.

31. 메뉴에서 **File > New From Template > Python File** 을 선택합니다.
32. 템플릿 파일에서 샘플 코드를 삭제합니다.
33. **File > Save As...**를 선택하고, 연습 파일에 적절한 이름(예: *debug-caesar-4.py*)을 입력한 다음 **/home/ec2-user/environment** 디렉터리에 저장합니다.
34. 다음 내용을 이 파일에 복사합니다.

```
# Module Lab: Caesar Cipher Program Bug #4
#
# In a previous lab, you created a Caesar cipher program. This version of
# the program is buggy. Use a debugger to find the bug and fix it.

# Double the given alphabet
def getDoubleAlphabet(alphabet):
    doubleAlphabet = alphabet + alphabet
    return doubleAlphabet

# Get a message to encrypt
def getMessage():
    stringToEncrypt = input("Please enter a message to encrypt: ")
```

```
    return stringToEncrypt

# Get a cipher key
def getCipherKey():
    shiftAmount = input("Please enter a key (whole number from 1-25): ")
    return shiftAmount

# Encrypt message
def encryptMessage(message, cipherKey, alphabet):
    encryptedMessage = ""
    uppercaseMessage = ""
    uppercaseMessage = message.upper()
    for currentCharacter in uppercaseMessage:
        position = alphabet.find(currentCharacter)
        newPosition = position + int(cipherKey)
        if currentCharacter in alphabet:
            encryptedMessage = encryptedMessage + alphabet[newPosition]
        else:
            encryptedMessage = encryptedMessage + currentCharacter
    return encryptedMessage

# Decrypt message
def decryptMessage(message, cipherKey, alphabet):
    decryptKey = -1 * int(cipherKey)
    return encryptMessage(message, decryptKey, alphabet)

# Main program logic
def runCaesarCipherProgram():
    myAlphabet="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    print(f'Alphabet: {myAlphabet}')
    myAlphabet2 = getDoubleAlphabet(myAlphabet)
    print(f'Alphabet2: {myAlphabet2}')
    myMessage = getMessage()
    print(myMessage)
    myCipherKey = getCipherKey()
    print(myCipherKey)
    myEncryptedMessage = encryptMessage(myMessage, myCipherKey, myAlphabet2)
```

```
    print(f'Encrypted Message: {myEncryptedMessage}')
    myDecryptedMessage = decryptMessage(myEncryptedMessage, myCipherKey,
myAlphabet2)
    print(f'Decrypted Message: {myEncryptedMessage}')

# Main logic
runCaesarCipherProgram()
```

35. 파일을 저장합니다.
36. 버그가 있는 네 번째 시저 암호 프로그램 실행을 시도합니다. 출력은 다음 예시와 비슷합니다.

```
Alphabet: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Alphabet2: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ
Please enter a message to encrypt: AWS Restart rocks!
AWS Restart rocks!
Please enter a key (whole number from 1-25): 2
2
Encrypted Message: CYU TGUVCTV TQEMU!
Decrypted Message: CYU TGUVCTV TQEMU!

Process exited with code: 0
```

37. 출력에 버그가 있는 것 같습니다. 마지막 버그를 찾아 수정합니다.

축하합니다! 4 가지 프로그램을 디버그하고 이 교육 과정의 모든 실습을 완료했습니다.

## 실습 종료

축하합니다! 실습을 마치셨습니다.

38. 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.

39. *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.