04 제어문(반복문)

- 주사위를 100번 던져 평균값을 구해야 한다면?
- 1부터 1000까지 더해야 한다면?
- 최근 1년의 주가 데이터를 가져와 분석해야 한다면?
- 지정한 횟수만큼 코드를 반복하게 만들어준다.
- 인간의 능력보다 빠르게 처리 가능

04 제어문(반복문)

- 1+2+3+4+ ..... + 1000
- [96, 44, 32, 29, 89, 91, 73, 64, 12, 100]

04 제어문(반복문_for)

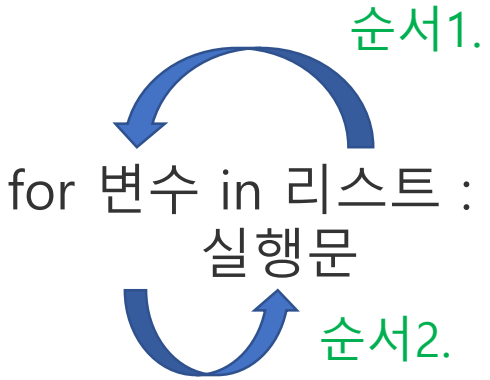
- for문의 기본 구조

  - ✓ for 변수 in 반복 가능한 객체&자료형(리스트, 튜플, 문자열 등) :

    실행문

  - ✓ 특정 자료의 값들을 하나씩 꺼내 변수에 담아 활용하는 컨셉

04 제어문(반복문_for)



- for문의 기본 구조

  ✓ 리스트에 있는 값들을 하나씩 꺼내 담을 때 마다 실행문 실행.

  ✓ 그리고 다시 리스트에 다른 값을 꺼낸다.

  ✓ 위 동작의 반복(반복 대상의 마지막 요소에 접근할 때까지)

04 제어문(반복문_for)

- 다른 언어(java)에서의 for문의 기본 구조

  ✓ for (int i=0, i<lista.length, i++):

  lista.get(i)

04 제어문(반복문_for)

- For문 예제를 통한 이해1

  ✓ str1 = 'for 반복문 연습'

```
for i in str1 :
        print(i)
```

04 제어문(반복문_for)

- For문 예제를 통한 이해2

  ✓ list_cookies = ['새우깡', '홈런볼', '초코파이']

  ```
  for cookie in list_cookies :

          print(cookie)
  ```

04 제어문(반복문_for)

- 변수의 범위

```
a=0

for a in [1,2,3] :

    print(a)

    a=0

    print(a)
```

04 제어문(반복문_for)

▪ for문 예제 01(list)

- ✓ 리스트 변수 list_num에 ‘one’, ‘two’, ‘three’ 값이 담겨있다.
- ✓ 하나씩 꺼내서 print하라.

04 제어문(반복문_for)

- for문의 다양한 형태

✓ for 변수 in [튜플로 구성된 리스트]

실행문

✓ 튜플 단위로 변수에 담는다.

04 제어문(반복문_for)

- 범위가 감당 범위를 벗어난다면?
  - ✓ 지금까지는 작성 가능한 범위였다.
  - ✓ 하지만 만번 이상, 백만번 이상 반복하는 프로그램이라면?
  - ✓ [1,2,3,4,~ 10000]이라고 작성해야 할까?

04 제어문(반복문_for)

- For문과 사용되는 range 함수

  - ✓ range(a, b) >> 의미 : a이상 b미만의 리스트
  - ✓ for i in range(a, b) : >> a부터 b-1까지의 숫자를 차례로 i에 담는다.

  - ✓ [예제]
  - ✓ for num in range(1, 101) : >> 슬라이싱 범위와 같은 개념
  - ✓ print(num)

04 제어문(반복문_for)

▪ for문 예제 02(추가적인 변수 사용법)

- ✓ for문과 range함수를 사용하여 1부터 100까지 더한 값을 출력하라.

04 제어문(반복문_for)

▪ for문 예제 03(구구단)

- ✓ for문과 range함수를 사용하여 원하는 단수의 구구단을 출력하는 프로그램을 작성하라.
- ✓ input함수를 통해 단수를 입력 받아 해결하라

```
C:\chi_py_project>"C:/Program Files/Python310/
단수를 입력하세요:3
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27
```

```
C:\chi_py_project>"C:/Program Files/Python310/
단수를 입력하세요:11
11 X 1 = 11
11 X 2 = 22
11 X 3 = 33
11 X 4 = 44
11 X 5 = 55
11 X 6 = 66
11 X 7 = 77
11 X 8 = 88
11 X 9 = 99
```

04 제어문(반복문_for)

▪ for문 예제 04(and 활용)

- ✓ 1~100 사이에서 3의 배수이면서 2의 배수가 아닌 수를 출력하라
- ✓ 그 합을 역시 출력하라.

[출력 화면]

3 9 15 21 … 99

누적합 : 867

04 제어문(반복문_for)

- 다른 언어(java)에서의 for문 range 구조

  ✓ for (int i=0, i>100, i++):

  print(i)

04 제어문(반복문_for)

- 반복문과 제어문의 조합

  ✓ 반복문과 제어문을 조합하면 더 섬세한 결과를 얻을 수 있다.

04 제어문(반복문_for)

▪ for문 예제 05(for&if 조합)

- ✓ 리스트 변수 list_score에 학생들의 점수 [90, 25, 67, 45, 80]가 담겨있다.
- ✓ For문을 이용하여 60점이 이상이라면 “n번째 학생은 합격입니다.”를
- ✓ 미만이라면 “불합격입니다.”를 출력하는 프로그램을 작성하시오

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_1_for.py
1번 학생 90점으로 합격입니다
2번 학생 25점으로 불합격입니다
3번 학생 67점으로 합격입니다
4번 학생 45점으로 불합격입니다
5번 학생 80점으로 합격입니다
```

04 제어문(반복문_for)

▪ for문 예제 06(약수 구하기)

- ✓ number라는 변수에 input함수로 하나의 정수를 입력 받는다.
- ✓ 해당값의 약수를 출력하시오
- ✓ 약수란 어떤 정수(number)를 나머지 없이 나눌 수 있는 정수

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/sample_code_01/Day07/loop_quiz/quiz01.py
수를 입력하세요:8
1 2 4 8
```

04 제어문(반복문_for)

- Continue문
  - ✓ 특정조건을 만나면 건너뛰고 다음 반복작업으로 넘어가라
  - ✓ continue 선언 이후 하위 코딩을 무시하고 for문의 시작으로 돌아가고 싶을 때 사용
  - ✓ While문과 사용법 동일(뒤에서 다시 한번 학습)

04 제어문(반복문_for)

- Break문
  - ✓ 조건문과 반복문을 사용중 특정 조건시 반복을 강제 종료 시키고 싶을 때 사용
  - ✓ While문과 사용법 동일(뒤에서 다시 한번 학습)
  - ✓ [예시]
  - ✓ 10만번 주가 데이터를 가져와라 그 과정 중 어떠한 값을 받아오면 강제로 종료하라

04 제어문(반복문_for)

▪ for문 예제 07(break)

- ✓ list_blood_type = ['b', 'b', 'ab', 'a', 'b', 'b']
- ✓ 위와 같은 순서의 혈액형 순으로 고객이 이벤트를 대기중이다.
- ✓ 가장 먼저 뽑힌 a형 고객이 이벤트에 당첨되는 프로그램을 작성하라.

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_2_for_if.py
1번 고객님b형이시군요. 불발입니다.
2번 고객님b형이시군요. 불발입니다.
3번 고객님ab형이시군요. 불발입니다.
4번 고객님 a형이시군요. 당첨되셨습니다.
```

04 제어문(반복문_for)

- 리스트 내포(List Comprehension)
  - ✓ list 안에서 for와 if 구문을 사용하는 문법을 의미
  - ✓ 매우 직관적인 프로그래밍이 가능 하나 기존 사용법을 숙달 한 후 사용

04 제어문(반복문_for)

- 리스트 내포 기본 구조

  ✓ **변수** = [실행문 for **변수** in 열거형 객체(리스트, 튜플, range 등)]

  1) for문에서 열거형 객체의 원소 하나를 **변수**로 넘겨 받는다.

  2) **변수**에 할당된 값을 실행문으로 처리한다.

  3) 처리된 결과를 **변수**에 순차적으로 추가(append) 한다

04 제어문(반복문_for)

- 리스트 내포 기본 구조

  ✓ **변수** = [실행문 for **변수** in 열거형 객체 if 조건식]

  1) for문에서 열거형 객체의 원소 하나를 **변수**로 넘겨 받는다.

  2) **변수**에 할당된 값을 조건식을 사용해 비교 한다.

  3) 조건이 True면 **변수**에 할당된 값을 실행문으로 처리한다.

  3) 처리된 결과를 **변수**에 순차적으로 추가(append) 한다

04 제어문(반복문_for)

▪ for문 예제 08(리스트 내포)

✓ 1~100까지의 숫자 중 3의 배수를 리스트에 담아 해당 리스트를 출력하라.

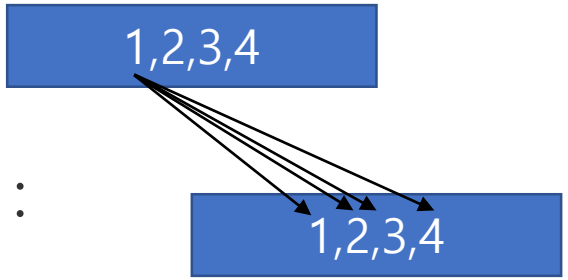
```
list_result = []
```

04 제어문(반복문_for)

- 2중 For문에 대해 알아보자. 그 이상의 for문도 가능

✓ for a in lista :

for b in listb :



✓ 두번째 for문은 첫번째 for문의 변수값이 달라질때마다 다시 시작.

04 제어문(반복문_for)

▪ 이중 for문 예제 09(구구단 출력)

- ✓ 구구단을 2단부터 9단까지 출력하라. 출력 형태는 아래와 같다
- ✓ 2x2 = 4
- ✓ 2x3 = 6
- ✓ …
- ✓ 9x9=81

04 제어문(반복문_for)

- For문을 활용하여 리스트에 각종 값을 담기 01

  ✓ 1~100까지의 숫자중 3의 배수를 리스트에 담은 후 해당 리스트를 print 하라.

  ✓ Hint 리스트에 담는다 = append함수 사용

04 제어문(반복문_for)

- For문을 활용하여 리스트에 각종 값을 담기 02
  - ✓ 구구단을 2단부터 9단까지 계산하여 모두 리스트에 담은 후
  - ✓ 해당 리스트를 print 하라.

04 제어문(반복문_for)

▪ for문 Final test

- ✓ list_position = ['c.과장', 'b.부장', 'd.대리', 'a.사장', 'd.대리', 'c.과장']
- ✓ 위 리스트를 대상으로 1)중복되지 않은 직급 2)직급별 인원 수를 dict형으로 출력하라
- ✓ 중복되지 않는 직급의 출력 순서는 무시해도 된다.

[출력 화면]

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_3_double_for.py
중복되지 않은 직급 : ['a.사장', 'b.부장', 'c.과장', 'd.대리']
각 직급별 인원수 : {'a.사장': 1, 'b.부장': 1, 'c.과장': 2, 'd.대리': 2}
```

04 제어문(반복문_while)

- For문과 while문의 차이

  - ✓ **For**문은 반복 횟수를 지정(정해진 반복 범위 만큼 실행)
  - ✓ **While**문은 조건을 지정

    조건을 지정해 두고 그 조건이 만족할 때(True)까지 계속 실행한다.

    따라서 조건을 False로 만드는 증가, 감소(증감) 문장이 필요

    만약 조건을 변경하는 문장을 입력하지 않으면 어떻게 될까?

04 제어문(반복문_while)

- while문의 기본구조

  ✓ while 조건문 : >> True면 반복, False면 종료

  실행문

  조건을 변화시키는 문장(증감식)

04 제어문(반복문_while)

- while문 예제를 통한 이해1

  ✓ count = 1

  while count < 5 :

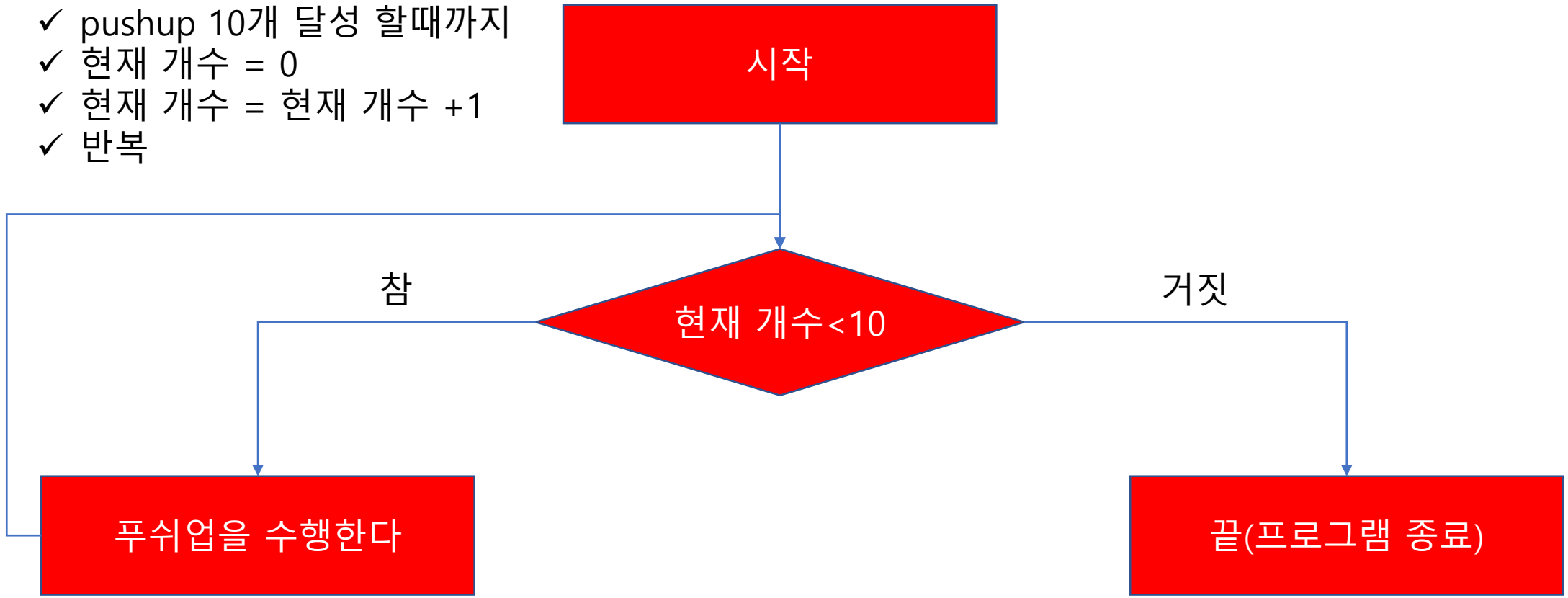
  print(count)

  count += 1

04 제어문(반복문_while)

▪ while문 예제 01

- ✓ pushup 10개 달성 할때까지
- ✓ 현재 개수 = 0
- ✓ 현재 개수 = 현재 개수 +1
- ✓ 반복



04 제어문(반복문_while)

- 출력은 아래와 같다.

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
푸쉬업 1회 수행했습니다
푸쉬업 2회 수행했습니다
푸쉬업 3회 수행했습니다
푸쉬업 4회 수행했습니다
푸쉬업 5회 수행했습니다
푸쉬업 6회 수행했습니다
푸쉬업 7회 수행했습니다
푸쉬업 8회 수행했습니다
푸쉬업 9회 수행했습니다
푸쉬업 10회 수행했습니다
오늘 운동을 끝마칩니다.
```

04 제어문(반복문_while)

▪ while문 예제02(1~100까지의 합)

- ✓ While문을 이용하여 1부터 100까지 더하는 프로그램을 작성하라
- ✓ 숫자는 키보드로 입력 받지 않고 num = 0 이라 변수 할당.

- ✓ [결과화면]

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
합산 결과는 다음과 같습니다. 5050
```

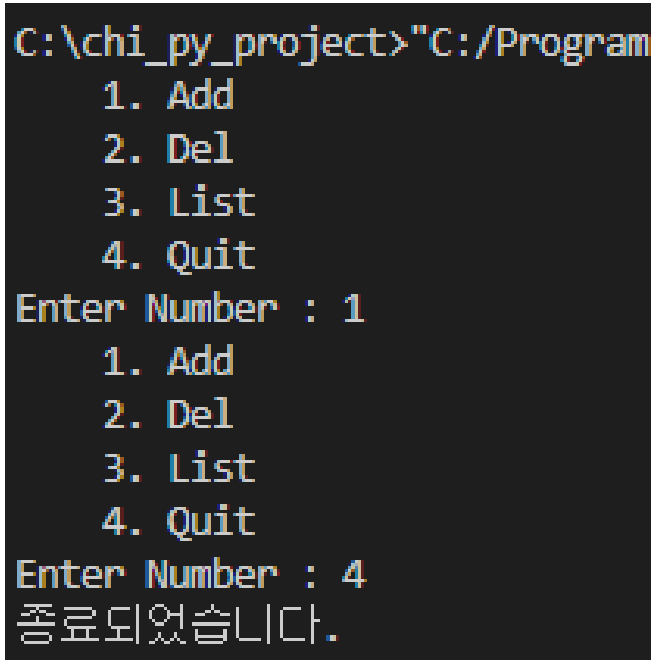
04 제어문(반복문_while)

▪ while문 예제03(문자열 다루기)

- ✓ 아래와 같은 결과화면이 나오게 프로그래밍 하여라.
- ✓ 단, 4를 입력하면 프로그램은 종료된다.
- ✓ msg = '''

  1. Add
  2. Del
  3. List
  4. Quit'''

[결과화면]



04 제어문(반복문_while)

▪ while문 예제04(나만의 리스트 만들기)

✓ 리스트의 크기를 키보드로 입력받아 그 크기만큼 임의 숫자를 리스트에 추가하고, 리스트의 크기와 값 전체를 출력하라. 모든 값은 키보드로 입력을 받고, list의 크기는 함수를 통해 구하라. 단, 리스트의 크기는 10 이하로 입력하라.

✓ [출력화면 예시]

```
C:\chi py project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
리스트의 크기를 정하세요!20
리스트의 크기를 10 이하로 다시 할당하세요!
리스트의 크기를 정하세요!4
리스트에 값을 할당해 보세요! 1
리스트에 값을 할당해 보세요! 2
리스트에 값을 할당해 보세요! 3
리스트에 값을 할당해 보세요! 4
크기4의 리스트 ['1', '2', '3', '4'] 가 할당 되었어요
```

04 제어문(반복문_while)

- Break문
  - ✓ 조건문과 반복문을 사용중 특정 조건시 반복을 강제 종료 시키고 싶을 때 사용
  - ✓ [예시]
  - ✓ 10만번 주가 데이터를 가져와라 그 과정 중 어떠한 값을 받아오면 강제로 종료하라

04 제어문(반복문_while)

▪ while문 예제05(break)

- ✓ 1~100까지의 합을 출력하라.
- ✓ 단, while 조건문은 아래 형태이고, break문을 사용하여 프로그래밍 하라.
- ✓ while True:
- ✓ Hint1 -> print(f"1부터 {num}까지의 덧셈 종료")

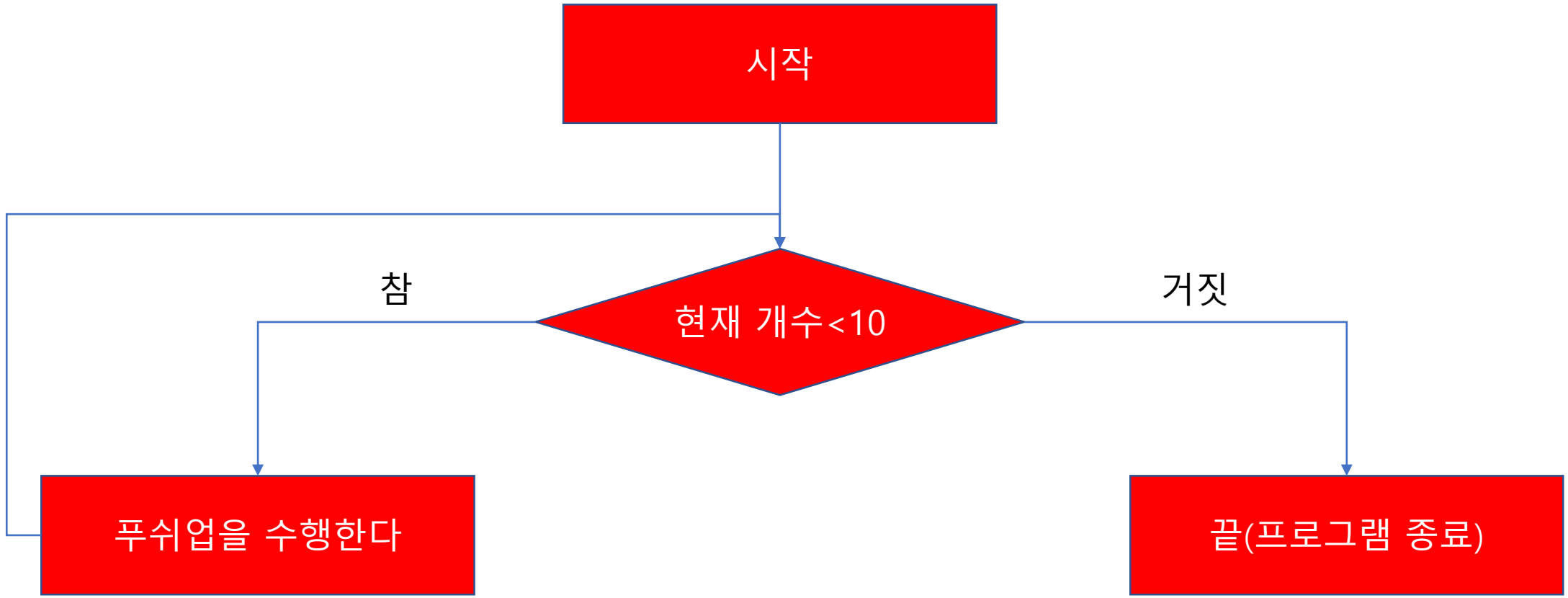
[결과화면 예시]

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
1부터 100까지의 덧셈 종료
결과 : 5050
```

04 제어문(반복문_while)

▪ while문 예제06(break)

✓ 푸쉬업 예제를 while True : 형태와 break를 사용해서 출력



04 제어문(반복문_while)

- Continue문
  - ✓ 특정조건을 만나면 건너뛰고 다음 반복작업으로 넘어가라
  - ✓ continue 선언 이후 하위 코딩을 무시하고 While문의 시작으로 돌아가고 싶을 때 사용

04 제어문(반복문_while)

▪ while문 예제07(continue)

✓ Continue를 사용해서 1~100 중 홀수만 출력하라

[결과 화면]

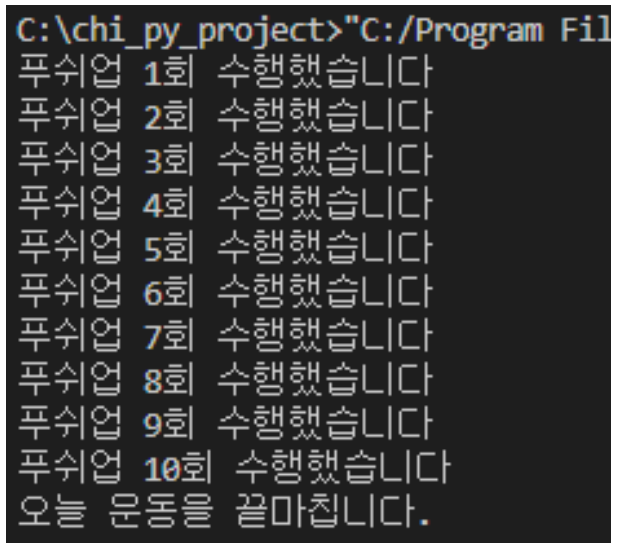
```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

04 제어문(반복문_while)

▪ while문 예제08(continue - break)

✓ 푸쉬업 예제를 break와 continue 모두를 사용해서 출력

[결과 화면]



04 제어문(반복문_while)

▪ while문 Final test01(숫자 맞추기)

- ✓ 1~10 사이의 난수 하나를 발생시켜 변수에 담고, 예상되는 숫자를 입력하여 맞추는 프로그램을 작성하라. 값이 일치하면 '성공', 입력한 값이 난수보다 크면 '더 작은 값를 입력하세요' 난수보다 작으면 '더 큰 값을 입력하세요' 라고 출력하라.

- ✓ [hint]
- ✓ import random
- ✓ num_random = random.randint(1,10) # 1~10 사이의 난수 발생

04 제어문(반복문_while)

▪ while문 Final test02

✓ 아래의 문자열 객체를 이용하여 단어를 추출하고 단어의 개수를 출력하라.

msg = '''비 갠 뒤에 비애 대신 a happy end

비스듬히 씩 비웃듯 칠색 무늬의 무지개

철없이 철 지나 철들지 못해 (still)'''

[화면출력 예시]

```
C:\chi_py_project>"C:/Program Files/Python310/python.exe" c:/chi_py_project/Day04/lecture04_4_while.py
['비', '갠', '뒤에', '비애', '대신', 'a', 'happy', 'end', '비스듬히', '씩', '비웃듯', '칠색', '무늬의', '무지개', '철없이', '철', '지나', '철들지', '못해', '(still)']
20
비
갠
뒤에
비애
대신
지나
철들지
못해
(still)
단어 개수 :  20
```

04_별첨 알고리즘

- 어떤 문제를 해결하기 위한 일련의 절차.
- 흔히 말하는 프로그램의 로직(Logic)

04_별첨 알고리즘(최대/최소값)

- 최대값 최소값을 구하라.
  - ✓ 1~100의 정수중 10개를 뽑아 리스트로 담고 최대값과 최소값을 구하라
  - ✓ [hint]
  - ✓ 1. num_random = random.randint(1,100) # 1~100 사이의 난수 발생
  - ✓ 2.
  - ✓ 3.

[출력화면 예시]

[57, 9, 27, 34, 58, 79, 61, 64, 15, 98]

최대값 = 98, 최소값 = 9

04_별첨 알고리즘(선택정렬[selection sort])

- 선택정렬 알고리즘은 특정 요소(값)을 기준으로 나머지 모든 원소를 비교해 가며 정렬하는 방법이다.

[출력화면 예시]

초기 리스트 : [5, 1, 3, 7, 2, 9]

1번째 정렬 : [1, 5, 3, 7, 2, 9]

2번째 정렬 : [1, 2, 5, 7, 3, 9]

3번째 정렬 : [1, 2, 3, 7, 5, 9]

4번째 정렬 : [1, 2, 3, 5, 7, 9]

5번째 정렬 : [1, 2, 3, 5, 7, 9]