CHAPTER 16

성적 처리기

학습 목표

기본 학습 목표

- 학생 클래스를 만들 수 있다.
- 학생 객체를 생성할 수 있다.
- __init()__ 생성자를 사용하여 객체 생성시 초기 값을 설정할 수 있다.

심화 학습 목표

- 객체 속성 값을 적절하게 접근하고 변경할 수 있다.
- 학생 클래스에 성적 등급을 부여하는 새로운 메소드를 추가할 수 있다.

핵심 학습 요소

핵심 학습 요소

- 클래스, 객체, 객체 생성자 `__init__()`

문제 상황

문제 상황 CT

학생의 성적을 처리하고자 한다. 학생의 성적은 중간고사, 기말고사, 과제에 의해 결정된다. 학생별로 학생 이름, 중간고사, 기말고사, 과제 성적을 입력받아 학생이 얻은 점수의 합과 평균을 구해서 알려주고 싶다.

문제 분석

문제 분석 CT

입력

- 학생 이름을 문자열로 입력
- 중간고사, 기말고사, 과제 성적을 정수형으로 입력

출력

- 학생 이름
- 계산된 합계
- 계산된 평균

문제 분석

문제 분해

- 학생 클래스 정의하기
- 성적 입력하고 학생 객체 생성하기
- 합계와 평균 구하고 보여주기

데이터

- 학생 이름을 나타내는 임의의 길이의 문자열
- 0부터 100까지 점수를 나타내는 정수
- 0부터 300까지의 합계를 나타내는 정수
- 0.0부터 100.0까지의 평균을 나타내는 실수

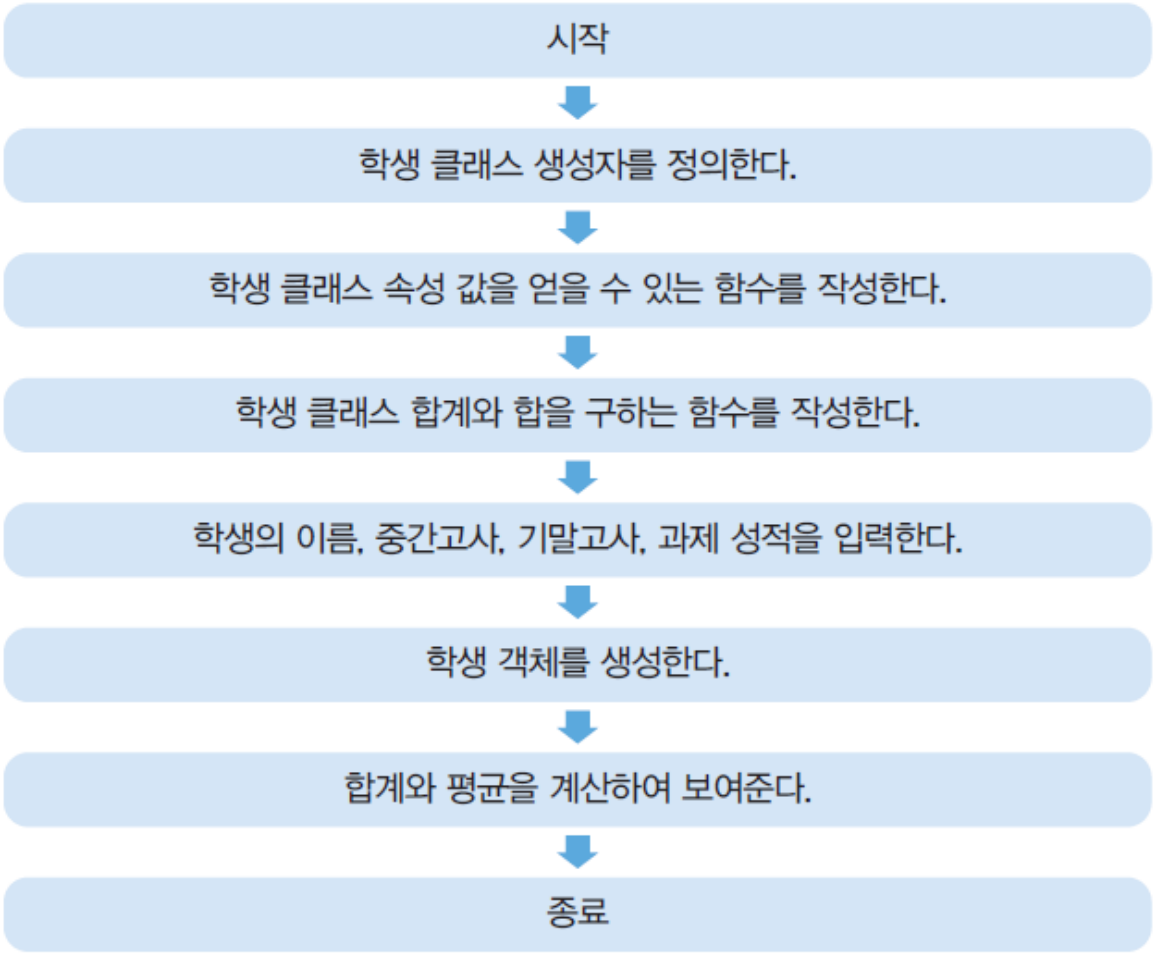
테스트와 디버깅

테스트와 디버깅 CT

입력값/출력 결과	확인 및 유의사항
이름 입력 : 홍길동 중간 고사 성적 입력 : 60 기말 고사 성적 입력 : 70 과제 성적 입력 : 80 학생 이름= 홍길동 합계= 210 평균= 70.0	▪ 예상된 테스트 결과인지 확인한다. ▪ 합계가 예상과 다른 경우 클래스내 calculate()가 제대로 작성되었는 지 확인한다. ▪ 객체 생성을 위한 26번째 라인이 제대로 실행되는 지 print() 함수를 이용하여 알아본다.

알고리즘

- 전체 알고리즘 뼈대



프로그래밍

변수

- name: 학생 이름
- midScore, finalScore, projectScore: 학생 점수
- student1: 학생 객체

프로그래밍

- 학생 클래스 정의
  - __init__()를 이용하여 초기화
  - 속성 값을 알려주는 메소드(함수) 작성
  - 합과 평균 메소드 작성(함수)

```
class Student:
    def __init__(self, name, midScore, finalScore, projectScore):
        self.name = name
        self.midScore = midScore
        self.finalScore = finalScore
        self.projectScore = projectScore

    def get_name(self):
        return self.name

    def get_sum(self):
        return self.sum

    def get_avg(self):
        return self.avg

    def calculate(self):
        self.sum = self.midScore + self.finalScore + self.projectScore
        self.avg = self.sum/3
```

프로그래밍

- 학생의 이름, 중간고사, 기말고사 입력 받음
- 객체 student1 생성

```
name = input('이름 입력 : ')

midScore =  int(input('중간 고사 성적 입력 : '))
finalScore =  int(input('기말 고사 성적 입력 : '))
projectScore = int(input('과제 성적 입력 : '))

student1 = Student(name, midScore, finalScore, projectScore)
```

실습해보기

실습해보기 16-1

학생 2명에 대해 객체를 만들도록 수정해보자.

프로그래밍

- 생성 객체인 student1을 이용하여 calculate() 호출
- get() 메소드와 set() 를 정의하여 속성 데이터 접근하거나 변경

```
student1.calculate()

print("\n학생 이름=", student1.get_name())
print("합계=", student1.get_sum())
print("평균=", student1.get_avg())
```

묻고 답하기

Q 묻고 답하기

문 학생 이름을 메소드 get_name() 대신에 student1.name과 같이 직접 접근해서 값을 가져오는 방법은 안되는가?

실습해보기

실습해보기 16-2

메소드를 사용하지 않고 직접 속성 값을 접근하여 동일한 출력 결과를 얻도록 수정해 보자.

실습해보기 16-3

student2에 대해서도 출력하도록 수정해 보자.

프로그래밍 – 전체 프로그램

- 1-18번째 줄 - Student 클래스를 정의

```
01  class Student:     # 클래스 정의
02      def __init__(self, name, midScore, finalScore, projectScore):
03          self.name = name
04          self.midScore = midScore
05          self.finalScore = finalScore
06          self.projectScore = projectScore
07
08      def get_name(self):
09          return self.name
10
11     def get_sum(self):
12          return self.sum
13
14      def get_avg(self):
15          return self.avg
```

프로그래밍 – 전체 프로그램

- 20-24번째 줄 - 학생의 속성 값을 입력 함수를 통해 입력 받음
- 26번째 줄 - 객체를 생성
- 28번째 줄은 생성된 객체에 대해 메소드 calculate()를 호출

```
16
17      def calculate(self):    # 합과 평균 계산하는 메소드
18          self.sum = self.midScore + self.finalScore + self.projectScore
19          self.avg = self.sum/3
20
21  name = input('이름 입력 : ')
22
23  midScore =  int(input('중간 고사 성적 입력 : '))
24  finalScore =  int(input('기말 고사 성적 입력 : '))
25  projectScore = int(input('과제 성적 입력 : '))
26
27  student1 = Student(name, midScore, finalScore, projectScore)  # 객체 생성
28
29  student1.calculate()      #합과 평균 계산 메소드 호출
30
31  print("\n학생 이름=", student1.get_name())
32  print("합계=", student1.get_sum())
33  print("평균=", student1.get_avg())
```

심화활동

심화 활동 CT

성적에 대해 A, B, C, D, F로 등급을 매기도록 수정해 보자. 단, A 등급은 90점 이상, B 등급은 80이상이고 90미만, C 등급은 70이상 80미만, D 등급은 60이상 70미만, F 등급은 60미만이다.