**1.시작하기에 앞서**

**1) 프로그래밍이란?**

- 프로그램: 특정 목적을 위해 컴퓨터에 내리는 명령의 집합

- 프로그래밍: 프로그램을 만드는 행위, 흔히 개발이라고 말함

- 코딩: 컴퓨터 언어로 작성하는 논리적인 ‘코드’를 작성하는 과정

**2) 파이썬이란?**

- 다른 언어에 비해 비교적 쉽고 높은 생산성을 가지고 있는 프로그래밍 언어

- 파이썬을 할 수 있는 것

① 웹프로그래밍: 여러 웹 서비스를 개발하는 것, 파이썬으로 웹프로그래밍을 할 때는 주로 장고라는 오픈소스 기반의 웹 프레임워크를 사용

② 빅 데이터 분석: 대용량 데이터를 분석하고 데이터 간의 패턴을 알려주며 이를 시각화하여 보여주는 분야, 인공지능 개발에 많이 사용

- 인터프리터 언어: 파이썬은 인터프리터 언어

① 인터프리터: 소스코드를 한 줄 씩 읽어 들이며 번역과 실행을 동시에 진행, 별도의 실행파일이 존재하지 않으며, 잘못된 명령을 수정하기 쉽다/ 파이썬, 자바스크립트 .

② 컴파일러: 소스 코드를 목적 코드로 번역하여 옮기기 때문에 실행 가능한 프로그램을 생성, 생성된 파프로그램은 목적 프로그램 혹은 바이너리 파일이라고 부름 /c언어, 자바

- 인덴트: 파이썬은 다른 언어에 비해 인덴트(공백)에 매우 민감한 언어

**3) IDE**

- 프로그래밍을 하기 위해 필요한 통합개발환경인 IDE에는 코드 작성 및 편집, 디버깅(논리적 오류인 버그를 검출 및 제거하는 과정), 컴파일(작성한 코드를 기계어로 번역하는 작업) 등의 기능을 갖추고 있다

**4) 파이썬 맛보기- 반복문**

- 순서도: 어떤 일을 처리할 때 여러 종류의 상자와 이를 잇는 화살표로 명령을 어떤 순서로 어떻게 처리해야 하는지 보여주는 도표

- 반복문은 일을 되풀이할 때 사용

**5) 파이썬 맛보기- 조건문**

- 조건문은 조건식의 참, 거짓을 판별하여 그에 맞는 결과를 실행

- 주석: 일종의 부가 설명으로 한 줄 주석 처리= #, 여러 문장 주석 처리=”””~”””(큰따옴표 세 개)

2. 변수와 콘솔 출력 함수

**1) 콘솔 출력의 기본**

- 콘솔: 컴퓨터와 소통하는 창

- 입력 값을 넣을 때 콘솔 입력 창 사용, 결과값을 확인할 때 콘솔 출력 창 사용

- 콘솔 출력: print(출력하고자 하는 데이터) 형태 ex) print(10)

**2) 변수의 선언과 초기화**

- 변수: 변하는 수, 특정한 형식을 가진 값을 저장하는 공간

- 변수의 세가지 요소

① 자료형: 변수가 가지는 특정한 형식(정수형, 실수형 등등)

② 변수명: 변수의 이름

③ 변수값: 변수에 저장한 값

- 변수 초기화: 새로 생성된 변수에 처음 값을 입력하는 과정

- 파이썬은 자료형 입력이 필요하지 않음 →파이썬은 자료형을 자동으로 추론하기 때문

- 동일한 자료형을 가진 변수는 한 문장으로 초기화 가능

- 변수는 변수로 초기화 가능 ex) (변수)e = (변수)a

스크린샷, 모니터, 화면, 노트북이(가) 표시된 사진

자동 생성된 설명

3. 기본적인 자료형과 연산자

**1) 콘솔 입력의 기본**

- 콘솔 입력 함수: input() 형태, 사용자가 콘솔 입력창에 직접 입력한 값을 변수에 할당

ex) a= input()

- input()은 입력한 값을 문자열로 저장 → 숫자를 입력하더라도 문자열로 인식

- 입력 받은 값을 숫자로 바꾸고 싶으면 int(), float() 사용

ex) a= int(input()) / a= float(input())

- 변수의 자료형을 출력을 통해 확인하고 싶다면 type() 함수 사용

ex) print(type(a))

**2) 숫자형**

① 정수 ex) 1, 2, 0

- 프로그램에서 int형으로 정의

② 실수 ex) 1.5, 2.56

- 프로그램에서 float형으로 정의

③ 복소수 ex) j, 3+2j

- 프로그램에서 complex형으로 정의

- 실수부= 변수.real, 허수부=변수.imag, 켤레복소수=변수.conjugate()

④ 16진수 ex) 0xDA

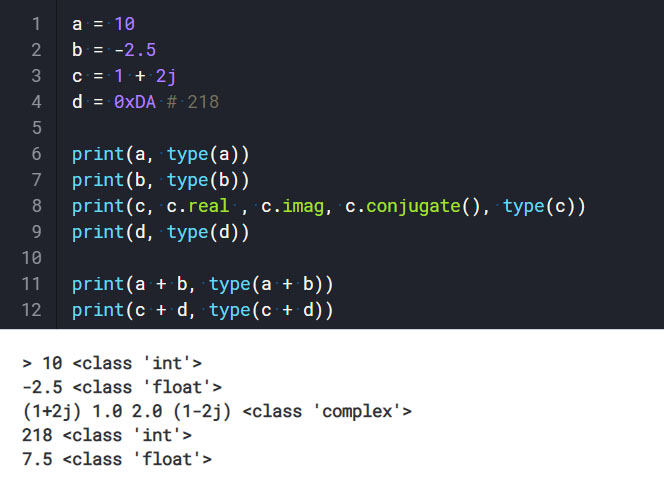
- 0x1A형태로 표현한다

- 형 변환없이 연산 가능하며 10진수로 출력된다

⑤ 2진수 ex) 0b110101

- 1, 0으로 표현한다

- 형 변환없이 연산 가능하며 10진수로 출력된다

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**3) 불(bool)형**

- bool형은 참, 거짓을 다루며 조건문과 같이 다닌다

- bool형 초기화 방식: 변수=True/False (앞글자는 반드시 대문자)

****

- bool형의 True/False는 주어진 조건에 대한 참/거짓뿐만 아니라 값의 존재 여부에 따라서도 판별

→ 값이 존재하면 참, 존재하지 않으면 거짓으로 판별

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**스크린샷이(가) 표시된 사진

자동 생성된 설명

**4) 수식 연산자: 두 개의 피연사자를 요구하는 이항 연산자**

스크린샷이(가) 표시된 사진

자동 생성된 설명

- 문자열 + 숫자는 연산이 불가능하다

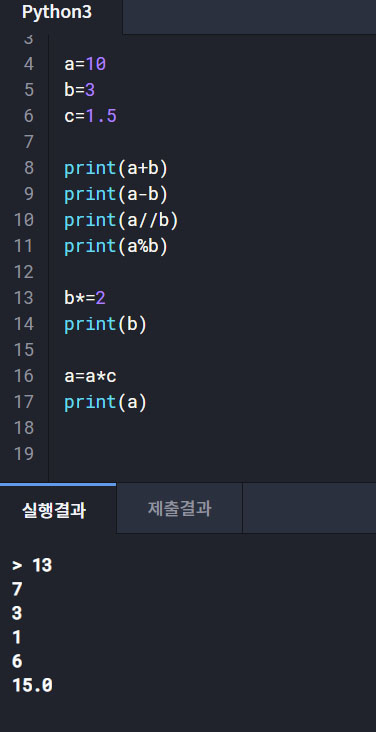
- 연산 대상 변수가 모두 숫자라면 int형이든 float형이든 자유롭게 연산 가능하다

**5) 할당 연산자**

- 할당 연산자: 변수에 값을 대입 혹은 할당할 때 쓰는 =

- 복합 대입 연산자: 할당연산자 = 와 수식 연산자를 함께 쓰는 것

**스크린샷이(가) 표시된 사진

자동 생성된 설명 **

**6) 논리 연산자: bool형 변수의 논리 연산을 위해 사용되는 연산자**

**스크린샷이(가) 표시된 사진

자동 생성된 설명** 텍스트, 모니터, 화면, 전화이(가) 표시된 사진

자동 생성된 설명

**7) 관계 연산자: 두 값의 대소 혹은 동등의 관계 따지는 연산자**

스크린샷이(가) 표시된 사진

자동 생성된 설명

- 출력결과가 어떤 숫자 값이 아니라 True, False로 나온다

**8) 비트 연산자**

스크린샷, 조류이(가) 표시된 사진

자동 생성된 설명

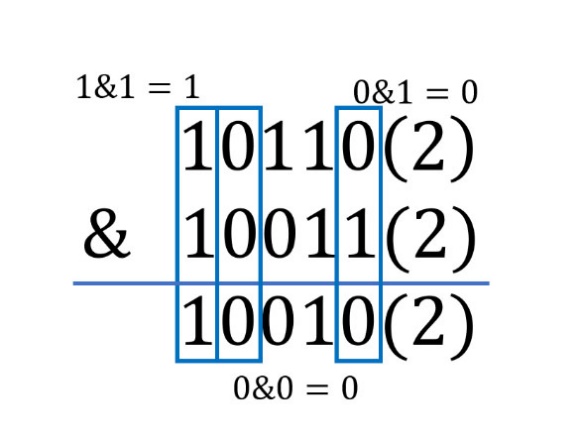
스크린샷, 조류이(가) 표시된 사진

자동 생성된 설명

- 모든 데이터를 2진수로 변환한 뒤 각 자리 수를 비트 연산한다

ex)

스크린샷이(가) 표시된 사진

자동 생성된 설명

→ a=22, b=19를 2진수를 변환하면 각각 10110(2), 10011(2)

그림 속 연산과정처럼 각 자리 수를 비트 연산하면 10010(2), 즉 18이라는 수가 나온다

4. 문자열

**1) 문자열 자료형**

- 문자열: 따옴표(작은 따옴표, 큰 따옴표)로 감싸져서 표현된 문자, 단어들의 묶음

- 문자열 표기 방식

① 작은 따옴표 ex) ‘안녕’

② 큰 따옴표 ex) “안녕”

③ 작은 따옴표 세 개로 감싸기 ex) ‘ ‘ ‘ 안녕’ ‘ ‘

④ 큰 따옴표 세 개로 감싸기 ex) “””안녕”””

→ 따옴표 세 개(작은&큰 따옴표)를 사용하면 명시적인 개행으로 문장을 나눌 수 있다

- 따옴표를 문자열로 표현하고 싶을 때

①작은 따옴표를 문자열로 표현: 큰 따옴표로 감싸고 문자열 안에 작은 따옴표 사용

ex) sentence= “그는 ‘안녕?’이라고 말했다”

②큰 따옴표를 문자열로 표현: 작은 따옴표로 감싸고 문자열 안에 작은 따옴표 사용

ex) sentence= ‘그는 “안녕?”이라고 말했다’

- 이스케이프 시퀀스: 문자열에서 표현하기 어려운 특정 문자를 백슬래쉬( | )로 표현하는 것

스크린샷, 화면, 모니터, 검은색이(가) 표시된 사진

자동 생성된 설명

→ |n 이외의 이스케이프 시퀀스는 자주 사용되지 않는다

**2) 문자열 연산**

- concatenation: 문자열을 덧셈 부호를 이용해 연결한 것, “문자열 \* 정수”의 형식

조류이(가) 표시된 사진

자동 생성된 설명

**3) 인덱싱과 슬라이싱**

- 문자열은 하나로 보일지라도 문자마다 인덱싱이 가능하다

- 인덱싱: 문자열에 포함된 각 문자에 대해 순차적으로 번호를 매기는 것

- 인덱스: 데이터 중 원하는 항목을 쉽게 찾도록 기록된 표기, 변수이름[인덱스값]의 형식

- 인덱스 값: 인덱스 번호

- 인덱스 번호는 0부터 시작

스크린샷, 시계이(가) 표시된 사진

자동 생성된 설명

- 공백도 인덱스에 포함되며, 앞에서뿐만 아니라 뒤에서부터도 접근 가능

→ 뒤에서 접근하면 마지막 요소는 -0이 아니라 -1부터 시작한다

시계이(가) 표시된 사진

자동 생성된 설명

- 문자열 자료형은 한번 초기화하면 사용자 임의로 값을 바꿀 수 없다

ex) str=”hello”일 때, str[2]=”u”라는 코드를 통해 바꿀 수 없다

- 슬라이싱: 단어 그대로 지정 범위만큼 데이터 요소를 잘라내는 기능

- 슬라이싱의 형식: 변수명[첫 인덱스 번호: 마지막 인덱스 번호]

- 슬라이싱할 때 인덱스 숫자 생략 가능

→ 앞 인덱스 값 생략하면 맨 처음부터/ 뒤 인덱스 값 생략하면 맨 끝까지

그리기이(가) 표시된 사진

자동 생성된 설명

- 슬라이싱 또한 – 를 붙임으로써 뒤 요소부터 접근 가능, 첫 인덱스값보다 마지막 인덱스값이 작으면 빈칸이 반환된다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

**스크린샷, 화면, 노트북, 전화이(가) 표시된 사진

자동 생성된 설명**

→ , 쓰면 저절로 띄어쓰기

**4) 문자열 포매팅1 ( % 키워드를 사용한 문자열 포매팅 )**

- 문자열 포매팅: 사용자가 문자열의 포맷을 지정하는 것

- 포맷 코드

스크린샷이(가) 표시된 사진

자동 생성된 설명

- 문자열 안에 %d, %s 같은 포맷코드 입력한 후, 문자열 뒤에 %를 시작으로 변수값을 작성한다

스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

\* 포맷 코드를 사용하지 않을 시

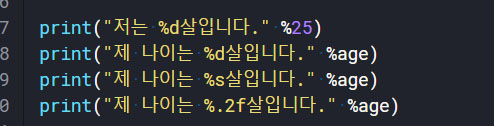
검은색, 전화이(가) 표시된 사진

자동 생성된 설명

- 숫자형은 문자열 포맷 코드를 사용하여 표현 가능, 문자열은 숫자형 포맷 코드 사용 불가능

그리기이(가) 표시된 사진

자동 생성된 설명

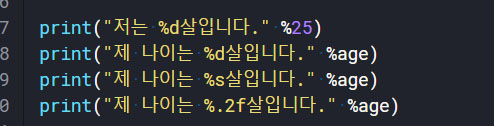


- %f 사용할 때, f 앞에 .(숫자)를 기록해 소수점을 몇 자리까지 표시할 것인지 지정 가능

텍스트, 그리기이(가) 표시된 사진

자동 생성된 설명

- 정수형은 실수형으로, 실수형은 정수형으로 포매팅 가능

텍스트, 그리기이(가) 표시된 사진

자동 생성된 설명

- 출력할 때 출력 폭 지정 가능

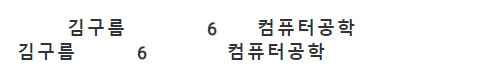
→ “%(폭)d” 형식으로 입력하면 자동으로 오른쪽 정렬, 왼쪽에는 공백이 채워진다

→ “%-(폭)d” 형식으로 입력하면 자동으로 왼쪽 정렬, 오른쪽에는 공백이 채워진다

→ “%숫자(폭)d” 형식으로 입력하면 빈 곳을 숫자로 채울 수 있다

측정기, 그리기이(가) 표시된 사진

자동 생성된 설명

**5) 문자열 포매팅2 (format() 함수를 사용한 포매팅)**

- {인덱스값}을 사용하여 format()함수 안의 값을 상관없이 사용

① 인덱스값을 바로 대입

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- {0}은 format(요소1, 요소2, …)이 때, 요소1의 값을 불러온다

- format() 함수를 이용한 포매팅 사용 형식: “문자열.format(요소1, 요소2, …)

② 변수 이름으로 대입

텍스트이(가) 표시된 사진

자동 생성된 설명

- **format() 함수 내에서 변수를 선언하고 초기화하면** 그 값을 “**{변수명}**” 형식으로 받아올 수 있다

- 이는 **인덱스 값과** 혼용하여 사용 가능

print("저의 이름은 **{1}**입니다. 그리고 나이는 **{age}**살이고 키는 **{0}**cm입니다. 제 가장 친한 친구는 **{name}**입니다.".format(**181.12, "김구름", height = 181.123, age = 25, name = "박에듀**"))

- format() 함수에서도 출력 폭 지정 가능

스크린샷이(가) 표시된 사진

자동 생성된 설명스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

- “{**인덱스값** or **변수**: (**출력형식**)}”. format(요소) 형식으로 사용

ex) print("{**length** **:** **<10**}" .format(**length=30**)) → 폭: 10, 정렬: 오른쪽 정렬

ex) print("{**length** **:** **0<10**}" .format(**length=30**)) → 폭: 10, 정렬: 오른쪽 정렬, 공백문자: 0

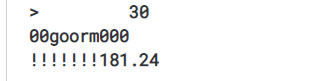
ex) print("{**0** **: 0^10**}" .format**("goorm"**)) → 문자열 “” 사용, 폭: 10, 정렬: 가운데 정렬, 공백문자: 0

ex) print("{**height** : **!>13**}" .format**(height=181.323**)) → 폭: 13, 정렬: 왼쪽 정렬, 공백문자: !

ex) print("{**height** : **!>13.2f**}" .format(**height=181.323**)) → 소수점 2자리 표시 , …

\* 다른 예시

텍스트이(가) 표시된 사진

자동 생성된 설명

③ f문자열 포매팅

- format() 함수와 마찬가지로 포맷 코드는 그대로 {} 사용하고 문자열 앞에 f만 붙여 사용한다

텍스트이(가) 표시된 사진

자동 생성된 설명

**6) 문자열 함수**

- 함수: 특정 기능을 수행하기 위해 제작된 명령 묶음, 함수 이름과 괄호를 기본으로 구성

- 전달 인자: 함수명 옆에 있는 괄호 안에 들어가는 값을 ‘전달 인자’ 라고 한다

ex) print(num) → num이 전달 인자

- 문자열 함수는 “문자열 or 문자변수. 함수()” 형식으로 사용

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

- len() 함수: 문자열뿐만 아니라 여러 자료형의 길이를 반환하는 함수, len(변수 or 값 자체) 형식

스크린샷, 그리기이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

<flexbox froggy>

