|  |  |
| --- | --- |
| 교육 제목 | **프로그래밍과 파이썬 기초** |
| 교육 일시 | 21년 09월 16일 |
| 교육 장소 | 영우글로벌러닝 - 비대면 수업(자택) |
| **교육 내용** | |
| 오전 | -컴퓨터 프로그램: 컴퓨터가 무엇을 해야 할지 미리 작성한 진행 계획  -이진 숫자:0과1로 이루어진 컴퓨터의 언어  -이진 코드(binary code): 이진 숫자로 이루어진 코드  -프로그래밍 언어:사람이 이해하기 쉬운 언어로 프로그램을 만들기 위해 만들어짐  -소스 코드: 프로그래밍 언어로 작성한 프로그램  -코드 실행기: 프로그래밍 언어는 컴퓨터가 이해할 수 없으므로 이를 이진 숫자로 변환해 주는 역할을 한다.  -파이썬: 비전공자도 쉽게 배울 수 있으며 다양한 분야에서 활용 할 수 있으며 대부분의 os에서 동일하게 사용되지만 인터프리터 언어이기때문에 c언어와 같은 컴파일러 언어에 비해 일반적으로 10~350배 느림, 하지만 컴퓨터 성능이 좋아지면서 연산이 많이 필요한 프로그램이 아니라면 차이를 크게 느낄 수 없음, 연산이 많이 필요한 경우라면 그 부분만 컴파일러 언어로 만들고 파이썬과 같이 연동해서 사용하기도 한다.  -개발환경: 프로그래밍을 할 수 있는 환경  -텍스트 에디터:긴 코드를  -인터프리터: 파이썬으로 작성된 코드를 실행해 주는 프로그램  -프롬프트: 코드를 한줄씩 입력하고 인터렉티브 셀은 곧 대화형 셀이고  -표현식:값을 만들어 내는 코드, 값은 숫자, 수식과 문자열 등이 될 수 있다.  -문장:표현식이 하나 이상 모인 것이지만 그 자체로 어떠한 값을 만들 수 없으면 문장이 될 수 없다.  -프로그램:문장이 모인 것.  -키워드: 특별한 의미가 부여된 단어이며 파이썬에서 이미 특정 의미로 사용하기로 예약해 놓은 것이며 사용자가 키워드랑 같은 단어로 식별자로 쓸 수 없다.  -식별자:프로그래밍 언어에서 이름 붙일 때 사용하는 단어이고 변수 또는 함수 이름 등으로 사용하고 키워드는 사용 못하고 특수문자는 언더바\_만 가능하며 숫자로 시작해서도 안되고 공백이 있어서도 안 된다. 그리고 알파벳을 쓰는 것이 관례이며 의미 있는 단어로 하는 것이 좋다.  -주석:프로그래밍 설명 시 사용하며 프로그램에는 영향을 끼치지 않는다, 파이썬에서는 # 뒤에 설명을 붙이면 된다.  \*스네이크 케이스: 단어 사이에 \_를 넣어 사용하는 식별자, 변수로 사용하거나 뒤에 괄호를 넣고 함수로 사용하면 좋다(안해도 상관은 없음).  \*캐멀 케이스: 단어 첫 글자를 대문자로 만든 경우이고 대부분의 경우 클래스 이름을 정할때 사용한다.  -print()함수: 출력하고 싶은 것들을 괄호 안에 나열하면 출력 된다.  \*프린트 함수의 괄호안에 아무것도 없으면 빈칸이 출력되고, 숫자는 그대로 출력이 가능하고 단어를 써 넣고 싶으면 “ ” 안에 말을 입력하면 출력이 가능하다.  -자료:프로그램이 처리할 수 있는 모든 것이고 프로그램은 자료를 처리하기 위한 모든 것이다.  -자료형:자료를 기능과 역할에 따라 구분한 것이다.  \*문자열(string):메일 제목, 메시지 내용 등의 글자들  1.큰따옴표와 작음따옴표를 이용해서 문자열을 사용할 수 있다.  2.따옴표를 문자열 안에 넣고 싶으면 서로 다른 따옴표를 넣으면 구문오류 없이 사용 가능하다, ex) print('”잘가.”라고 그는 말했습니다.')  3.이스케이프 문자(excape character\)를 이용해서 문자열을 사용할 수 있다. 문자열을 출력할때 따옴표를 넣을 수 있다, ex) print('\'잘 가.\''라고 그녀는 생각했습니다.) = '잘 가.'라고 그녀는 생각했습니다. 탭을 사용하려면 \t를 사용하면 된다.  ex) print("이름\t나이\t지역")  print("홍길동\t250\t개성")  print("고길동\t45\t쌍문동")  print("YGL\t28\t학동")  4.문자열을 한 문장에서 여러 줄을 만들 수도 있는데, \n을 이용해서 여러 줄로 만들 수도 있다. ex) print("Cogito\nergo\nsum") 그리고 큰따옴표와 작은따옴표를 세 번 반복해서 만들 수도 있다, ex) print("""Cogito  ergo  sum""")  줄바꿈 없이 문자열 만들기 또한 가능한데 세 번 반복한 따옴표들 사이에 \문자열\을 넣으면 사용 가능하다.  \*숫자(number): 물건의 가격, 학생의 성적 등의 숫자들  \*불(boolean):친구의 로인 상태 등의 상태를 나타낸다, True, False 첫글자는 무조건 대문자이다.  -type():자료의 형식을 확인하는 함수이다, print(type(”hello”))를하면 hello의 자료형인 문자열을 나타내는 단어가 print()함수에 의해 출력된다.  -문자열 연산자:문자열 사이에 +연산자로 문자열을 이을 수도 있다.  ex) “안녕”+”하세요” =안녕하세요  \*연사자를 문자열 앞에 넣어서 반복을 할 수도 있다.  ex) print(3\*"제발")  둘을 합쳐서도 사용 가능하다, ex) print(3\*"제발"+" 살려주세요") |
| 오후 | -문자 선택 연산자(인덱싱) [] : 문자열 내부의 문자 하나를 선택해서 대괄호 안에 선택할 문자의 위치를 지정한다  \*인덱스(index)  1. 제로 인덱스(zero index):숫자를 0부터 센다. 파이썬은 제로인덱스를 사용한다.  2. 원 인덱스(one index):숫자를 1부터 센다.  3.문자열 범위 선택 연산자(슬라이싱) -:- : 범위를 지정해 : 연산자로 문자열을 자를 수 있다 ex) '1:2' ->2번재부터3번째 위치의 연산자를 잘라 읽는다. ex) ':3' ->4번째까지 문자열을 읽는다, ex) '1:' ->2번째 문자열부터 읽는다.  -len()함수 :문자열 길이를 구할 때 사용하며 ex) print(len(”안녕하세요”)) 결과는 5가 출력된다 왜냐하면 안녕하세요가 5글자이기 때문  -정수형 : 1 2 3 4 5...  -실수형 : 1.1 ...  -숫자 연산자  \*사칙 연산자 + - \* /  1.+:알고있는 덧셈처럼 사용하면 된다  2.-:알고있는 뺄셈처럼 사용하면 된다  3.정수 나누기 연산자 // : //연산자는 나눗셈을 한 뒤 소수점 자리를 떼어버리고 정수만을 남긴다  4.나눗셈 연산자/ : / 알던 나눗셈을 하는데 파이썬은 float을 자동으로 읽어서 바꾼다  5.곱셈 연산자 \* :곱셈을 한다  6.몫 연산자 %: 나눗셈을 한뒤 나머지만 구한다  7.제곱 연산자 \*\*:제곱을 한다  -연산자의 우선순위 : 현실에서 하는 계산처럼 곱셈과 나눗셈이 뺄셈과 덧셈보다 우선순위에 있고 왼쪽부터 계산을 한다.  변수와 입력  -변수: 값을 저장할 때 사용하는 식별자며 숫자뿐만 아니라 모든 자료형을 저장할 수 있다. ex) pi = 3.14 print(pi) 결과는 3.14  -복합대입 연산자: 기본연산자=와 연산자함께 사용해서 구성한다  ex) number=100  number += 10  number +=20  print(”number:”,number)  결과 number:130  넘버는 100이고 거기에 10을 더한후 바로 넘버가 110으로 대입이 되고 거기에 20을 더하고 그걸 출력하니 130으로 나온다.  -문자열 복합 대입 연산자도 문자열끼리 같은 일을 할 수 있다.  -대입연산자: 변수에 값을 저장한 것 ex) a = 10; print(a) 결과 10  캐스트  -문자열을 수자로 바꾸기(input()의 경우 뭐를 입력하던 문자열이다)  \*int()함수: 문자열을 정수의 숫자로 바꿔준다  \*float()함수: 문자열 실수의 숫자로 바꿔준다  \*위의 두 함수 모두 변수가 숫자일 경우에만 작동을 하고 '안녕하세요'의 경우 숫자가 아니기 때문에 에러가 일어난다  \*str()함수:숫자를 문자열로 바꿔준다 ex) str(1234) 결과 '1234' 파이썬은 결과에 문자열이 나오면 작은따옴표로 문자열이라는것을 알려준다. 그러니 저 1234는 천이백삼십사가 아니라 '1234'라는 문자열인 것이다 실제로 str(1234) + str(5678)을 실행하면 1234+5678 의 연산결과가 아니라 '12345678'가 결과로 뜬다. |