<빠른 정답>

캘럿 내신대비 파이썬 프로그래밍 [문제편]

자동채점기 ➜

1차시	1	O	2	X	3	X	4	X	5	O	6	④	7	③
	8	②	9	④	10	③	11	해설 참조						
2차시	12	X	13	O	14	O	15	X	16	X	17	X	18	O
	19	X	20	O	21	O	22	①	23	②,⑤	24	③	25	③
	26	②	27	④	28	⑤	29	리스트 (또는) list			30	ⓐ 2.718	ⓑ 500	
	ⓒ float		31	③	32	③	33	②	34	⑤	35	④	36	1
	37	21	38	④	39	③	40	①	41	⑤	42	④	43	②
	44	④	45	②	46	해설 참조								
3차시	47	O	48	X	49	X	50	X	51	O	52	O	53	O
	54	⑤	55	③	56	①	57	②	58	③	59	⑤	60	④
	61	②	62	④	63	②	64	해설참조	65	해설참조	66	⑤	67	7777
	68	{int(num)**2}			69	해설참조	70	ⓐ total//60 ⓑ total%60			71	ⓐ total//3600		
	ⓑ (total%3600)//60 ⓒ total%60						72	age>=18	73	④	74	①, ②, ④		
	75	②	76	해설참조	77	②, ③, ④			78	①	79	③	80	③
4차시	81	O	82	X	83	O	84	O	85	X	86	X	87	X
	88	O	89	O	90	X	91	O	92	X	93	①	94	④
	95	④	96	③	97	②	98	③	99	④	100	④	101	④
	102	③	103	④	104	해설참조	105	A	106	거짓	107	PASS		
	108	③	109	B	110	②								
5차시	111	X	112	O	113	O	114	O	115	O	116	X	117	O
	118	O	119	⑤	120	②	121	③	122	④	123	③	124	①
	125	③	126	012	127	해설참조	128	해설참조	129	③	130	②	131	③
	132	④	133	⑤	134	③	135	①	136	해설참조	137	①	138	④
	139	②	140	④	141	ⓐ 6 ⓑ 1			142	18	143	②	144	⑤
	145	②	146	ⓐ "*" ⓑ dan*i										
6차시	147	X	148	X	149	O	150	X	151	X	152	X	153	④
	154	④	155	해설참조	156	ⓐ f.read() ⓑ f.close()			157	④	158	①	159	②
	160	②	161	⑤	162	④	163	①	164	③	165	③	166	④
	167	③	168	해설 참조			169	해설 참조						
7차시	170	O	171	O	172	O	173	O	174	X	175	X	176	O
	177	②	178	②	179	④	180	④	181	⑤	182	④	183	②
	184	①	185	12	186	127	187	②	188	③	189	1	190	5번줄
	191	②	192	해설참조	193	30	194	4	195	①	196	③	197	④
	198	③	199	⑤										
8차시	200	O	201	X	202	O	203	X	204	O	205	X	206	X
	207	O	208	b.append(i*2)			209	③	210	④	211	해설참조	212	⑤
	213	②	214	-15	215	④	216	25	217	③	218	②	219	해설참조
	220	②	221	ⓐ append ⓑ insert			222	[10, 30]			223	③	224	④
	225	③	226	[100, 10, 15, 200]			227	[3, 1, 4, [1, 4]]			228	해설 참조		
	229	[[10, 15, 20], [30]]		230	③		231	[10, 15, 30]			232	②		
	233	['grape', 'banana', 'melon']												
9차시	234	O	235	X	236	X	237	O	238	X	239	O	240	⑤
	241	①	242	②	243	strip	244	one,two,three			245	해설참조	246	②
	247	cd												
11차시	248	O	249	X	250	O	251	O	252	X	253	O	254	O
	255	④	256	③	257	⑤	258	④	259	⑤	260	③	261	동수
	262	20	263	8										

Ver 1.0

1차시 객체지향 프로그래밍과 파이썬 [개념편 6쪽 / 문제편 8쪽]

문제 1. 정답: ○ **| 해설:** 파이썬은 숫자, 문자열, 함수 등 모든 요소를 객체로 간주하는 언어이다.

문제 2. 정답: X **| 해설:** 일반적으로 인터프리터 언어는 컴파일러 언어보다 느리다.

문제 3. 정답: X **| 해설:** 객체는 데이터(속성)를 포함할 뿐 아니라 동작(기능)도 정의하여 실행된다.

문제 4. 정답: X **| 해설:** 컴파일러는 전체 코드를 한 번에 번역하고 실행한다.

문제 5. 정답: ○ **| 해설:** 절차적 프로그래밍은 기본적으로 순차, 조건, 반복 구조를 사용한다.

문제 6. 정답: ④ **| 해설:** [4] 객체는 데이터(속성)과 함수(기능, 동작)을 묶은 것이다. (○) [1] 데이터와 함수를 분리하는 것은 절차지향 프로그래밍이다. (X) [2] 객체지향 프로그래밍은 문제를 객체 단위로 쪼개고 객체들이 상호작용하며, 순차적인 실행 흐름이 아닌 객체 중심의 구조적 설계가 중심이다. (X) [3] 객체는 기능(함수) 뿐만 아니라 속성(데이터)도 가질 수 있다. (X) [5] 객체는 객체지향 프로그래밍의 핵심이다. (X)

문제 7. 정답: ③ **| 해설:** 객체지향은 객체 단위로 나눠 복잡한 문제를 쉽게 해결할 수 있도록 한다.

문제 8. 정답: ② **| 해설:** [2] '기능 중심', '위에서 아래로 순차적'은 절차적 프로그래밍의 특징이다. (X) [1] 객체를 재사용할 수 있다. (○) [3] 객체지향 프로그래밍으로 '사람', '자동차' 등 현실 세계의 개념을 객체로 모델링할 수 있다. (○) [4] 나중에 배울 정보 은닉-캡슐화를 통해 객체의 내부 구현을 숨기고, 외부에는 인터페이스만 제공할 수 있다. (○) [5] 이 책에서 다루지는 않지만, 객체지향의 다형성이라는 것은 같은 이름의 메서드가 다양한 객체에서 다르게 동작하게 하는 것으로, 대표적인 예로 __len__이 있다. (○) **아예 모르는 내용이 시험에 나온다면 가장 확실하게 정답인 것을 골라야 한다.**

문제 9. 정답: ④ **| 해설:** [ㄴ] 파이썬은 인터프리터 언어로, 코드를 한 줄씩 읽고 바로 실행한다. (○) [ㄷ] 파이썬은 문법이 간결하고 영어처럼 읽히기 때문에 초보자도 쉽게 배울 수 있으며, 웹 개발, 데이터 과학, 인공지능, 자동화 등 다양한 분야에서 활용된다. (○) [ㄱ] 파이썬은 객체지향을 포함하여 절차적 등 여러 프로그래밍 패러다임을 모두 지원하는 유연한 언어이다. (X)

문제 10. 정답: ③ **| 해설:** [ㄱ] 파이썬은 Windows, MacOS, Linux 등 여러 운영체제에서 동일한 코드로 실행할 수 있다. (○) [ㄴ] 파이썬은 고급 언어이자 가독성이 좋아 초보자에게도 적합한 문법 구조를 가지고 있다. (○) [ㄷ] 파이썬은 인터프리터 방식이며, 컴파일 방식으로만 실행되지 않는다. (X) [ㄹ] 웹 개발뿐 아니라 데이터 분석, 인공지능, 자동화 등 다양한 분야에 사용된다. (X) [ㅁ] NumPy, pandas, TensorFlow 등 외부 라이브러리를 통해 기능 확장이 가능하다. (○)

문제 11. 모범 답안: 파이썬은 영어처럼 읽히는 코드로 초보자도 쉽게 배울 수 있다. 또한 웹 개발, 데이터 과학, 인공지능, 자동화 등 많은 분야에서 사용할 수 있으며, NumPy, pandas, TensorFlow 등 다양한 라이브러리가 존재하여 기능 확장이 가능하다.

항목		기준	배점
1	초보자도 배우기 쉬운 문법	"영어처럼 읽힌다", "초보자도 쉽게 배울 수 있다" 등이 언급되었는가?	0.5점
2	다양한 활용 분야	"웹 개발", "데이터 과학", "AI", "자동화" 등이 언급되었는가?	1점
3	인터프리터 언어	"실행 결과를 바로 확인할 수 있다" 등이 언급되었는가?	1점
4	운영체제와 관련없이 실행	"운영체제와 관련없이 실행" 등이 언급되었는가?	0.5점
5	다양한 라이브러리	"NumPy", "pandas", "라이브러리", "패키지" 등이 언급되었는가?	1점
답에 항목이 2가지 이상 포함되었을 경우 4점, 한 가지만 명확하게 포함되었을 경우 배점 기준에 따름.			

2차시

변수와 자료형 & 연산 [개념편 8쪽 / 문제편 11쪽]

문제 12. 정답: X **| 해설:** 변수 이름은 숫자로 시작할 수 없으며 반드시 문자 또는 밑줄(_)로 시작해야 한다.

문제 13. 정답: O **| 해설:** 변수 이름에 공백을 포함할 수 없으며 공백 대신 밑줄(_)로 공백을 표현한다.

문제 14. 정답: O **| 해설:** 파이썬에서는 변수에 어떤 자료형이든 저장할 수 있으며, 이러한 특성을 '동적 타이핑'이라고 한다. 값을 변수에 넣으면 파이썬이 알아서 자료형을 판단한다.

문제 15. 정답: X **| 해설:** =은 저장(할당)에 사용하고, ==은 두 값이 같은지 비교할 때 사용한다.

문제 16. 정답: X **| 해설:** type()은 변수, 데이터의 자료형을 알려주는 함수이다. 변수의 값을 출력하는 함수는 print()이다.

문제 17. 정답: X **| 해설:** 파이썬에서 불리언 자료형은 True, False처럼 첫 글자를 대문자로 사용해야 한다.

문제 18. 정답: O **| 해설:** \n의 n은 newline의 약자로, 문자열에서 줄바꿈을 표현한다.

문제 19. 정답: X **| 해설:** %는 나머지를 구하는 연산자이므로, 7 % 2의 계산 결과는 1이다.

문제 20. 정답: O **| 해설:** `a *= 2`는 `a = a * 2`와 같은 의미이다.

문제 21. 정답: O **| 해설:** and는 양쪽의 두 조건이 모두 참일 때만 결과가 참이다.

문제 22. 정답: ① **| 해설:** 변수의 이름은 숫자로 시작할 수 없다.

문제 23. 정답: ②, ⑤ **| 해설:** 1 변수명에 대시(-)를 포함할 수 없다. (X) 3 & 4 파이썬 문법에 사용되는 단어인 예약어는 변수명으로 사용할 수 없다. (X) 예약어로는 if, else, pass, break, class, try, while, for, with, and, or, as, True, False, import, return 등이 있다. 이러한 **예약어는 지금 외우지 않아도 계속해서 파이썬을 배우면 자연스럽게 알게 된다. 지금 따로 시간 내서 외우지 말자.**

문제 24. 정답: ③ **| 해설:** True는 파이썬의 예약어이므로 변수 이름으로 사용할 수 없다.

문제 25. 정답: ③ **| 해설:** 1 변수명에 공백을 포함할 수 없다. (X) 2 변수명에 언더바(_) 외 특수 문자를 사용할 수 없다. (X) 4 if는 파이썬의 예약어로, 변수 이름으로 사용할 수 없다. (X) 5 list는 파이썬 내장함수 list()의 이름으로, 이를 변수 이름으로 사용하면 list() 함수의 기능이 사라지므로 변수명으로 사용하지 않는다. (X)

문제 26. 정답: ② **| 해설:** 학생 A 파이썬은 변수에 숫자, 문자열, 리스트 등 다양한 유형의 값을 자유롭게 할당할 수 있다. (O) 학생 B 파이썬에서는 변수를 선언할 때 자료형을 따로 지정해주지 않는다. 자료형을 지정해주지 않더라도, 변수에 어떠한 값을 대입하면 파이썬이 자동으로 어떤 자료형인지 판단한다. (X) 학생 C 파이썬에서는 문자열을 큰따옴표(") 뿐 아니라 작은따옴표(')로도 표현할 수 있다. 단, 큰따옴표로 열었으면 큰따옴표로 끝내고, 작은따옴표로 열었으면 작은따옴표로 끝내야 한다. 서로 다른 따옴표로 문장을 묶어서 문자열을 표현할 수는 없다. (X) 학생 D 파이썬에서는 변수에 값을 재할당하면서 자료형도 바꿀 수 있다. (X)

문제 27. 정답: ④ **| 해설:** "5"는 따옴표로 감싼 문자열이므로 문자열(str)이다.

문제 28. 정답: ⑤ **| 해설:** 파이썬에서 문자열은 따옴표로 감싸야 하며, 따옴표가 짝지어져 있어야 한다. 큰따옴표로 열었으면 큰따옴표로 끝내고, 작은따옴표로 열었으면 작은따옴표로 끝내야 한다는 것이다. 서로 다른 따옴표로 문장을 묶어서 문자열을 표현할 수는 없다.

문제 29. 정답: 리스트 (또는) list

문제 30. 정답: ⓐ 2.718 ⓑ 500 ⓒ float **| 해설:** 먼저, 우측 <결과>에 나오는 3줄의 결과는 각각 test1 변수의 값, test2 변수의 자료형, test2 변수의 값이다. test2는 처음에 ['Hello', 'World']라는 리스트였지만, 나중에 빈칸ⓐ로 변수에 값이 재할당 되었으므로 <결과>에 있는 2.718이 빈칸ⓐ에 들어가야 한다. 또, 2.718은 실수형 자료형이므로 float가 빈칸ⓒ에 들어가야 한다. 또, 빈칸ⓑ 또한 test1 변수가 재할당된 후 출력된 것이므로 마지막 할당값인 500이 출력되기에 500이 빈칸ⓑ에 들어간다.

문제 31. 정답: ③ **| 해설:** 할당 연산자는 = 기호로, 값을 변수에 할당(저장)할 때 사용된다.

문제 32. 정답: ③ **| 해설:** 3 변수명은 숫자로 시작할 수 없다.

문제 33. 정답: ② **| 해설:** 파이썬에서 자료형을 확인할 때는 type() 함수를 사용한다.

문제 34. 정답: ⑤ **| 해설:** 괄호의 위치를 잘 보고 판단하면 된다. 수학에서 평균을 구하는 방법과 같다.

문제 35. 정답: ④ **| 해설:** 1 var1 and var2 ➜ True and False ➜ **False**
2 var1 == var2 == var3 ➜ True == False == True ➜ False == True ➜ **False**
3 not var1 and var3 ➜ not True and True ➜ **False**
4 var1 != var2 ➜ True != False ➜ **True**
5 var2 == var3 ➜ False == True ➜ **False** and는 둘 다 참(True)이어야 참(True)이다.

문제 36. 정답: 1 **| 해설:** a ** b는 5의 2제곱이므로 25이고, 25 % 3은 25를 3으로 나눈 나머지이므로 변수 c의 값은 1, 따라서 print(c)는 1을 출력한다.

문제 37. 정답: 21 **| 해설:** 변수 x의 값은 10 ➜ 13(x 더하기 3) ➜ 26(x 곱하기 2) ➜ 21(x 빼기 5) 순서로 변화한다.

문제 38. 정답: ④ **| 해설:** 4 a += 1은 a = a + 1과 같은 의미의 복합 (할당) 연산자이다. (O) 1 변수명은 숫자로 시작할 수 없고, 특수문자는 언더바(_)를 제외하고 사용할 수 없다. (X) 2 문자열은 큰따옴표(" ") 또는 작은따옴표(' ') 모두 사용할 수 있다. 단, 따옴표로 묶을 때 섞어서 사용하는 것은 불가하다.(예: "hello') (X) 3 True는 불리언 자료형이고, 수학적으로는 1로 간주된다. False가 0이다. 5 파이썬에는 문자 하나를 위한 char 자료형이 없으며, 문자열(str) 자료형 하나로 문장과 문자 모두 표현한다. (X)

문제 39. 정답: ③ **| 해설:** 3 연산자 **는 거듭제곱 연산자로, 7**2는 7^2와 같다. (O) 1 연산자 //는 몫을 반환한다. 몫은 소수점이 있는 숫자인 실수일 수 없다. (X) 2 연산자 %는 나머지를 반환한다. 나머지도 몫처럼 정수만 가능하다. (X) 4 ==는 비교 연산자로, 두 값이 같은지 비교하여 True 또는 False를 반환한다. 값을 저장하는 연산자는 =이다. (X) 5 !=는 '같지 않음'을 의미하며, a와 b가 같을 때는 False, 다를 때는 True를 반환한다. (X)

문제 40. 정답: ① **| 해설:** 1 파이썬에서는 변수를 선언할 때 값도 바로 할당할 수 있으며, 할당된 값에 따라 파이썬이 자료형을 판단한다. (O) 2 =은 할당 연산자이고, ==는 비교 연산자(같은지 판단하는 연산자)이다. (X) 3 파이썬에서는 변수에 어떤 자료형이든 자유롭게 할당할 수 있으며, 재할당할 때에도 다른 자료형으로 바꿀 수 있다. (X) 4 7 / 2는 파이썬에서 실수 나눗셈으로 처리되어 3.5가

된다. 정수 나눗셈은 // 연산자이다. (X) [5] True와 1은 각각 불리언과 정수형이지만 조건문에서 동일하게 참으로 평가된다. (X)

문제 41. 정답: ⑤ **| 해설:** `w = x > 10 and z`는 논리 연산자 `and`를 사용한 표현으로, `x > 10`이 `True`일 경우 `z`의 값을 반환한다. 이때 `w`는 불리언(논리)이다. 두 값을 더한 것이 아니다.

문제 42. 정답: ④ **| 해설:** 처음 1~2줄에 의해 `a=10`, `b=3`이다. 3번 줄에 의해 `a`는 `a+2`이므로 `a=12`이다. 4번 줄에 의해 `b`는 `b×a`이므로 `3×12` 즉, `b=36`이다. 이후, 마지막 줄의 조건에서 `a>b`는 `12>36`이므로 `False`, `b%2==0`은 `36%2==0`인데, 36을 2로 나누었을 때의 나머지는 0이 맞으므로 `True`이다. 최종적으로, `False or True`가 된다. `or`는 둘 중 하나라도 참이면 참이므로 `result`에는 `True`가 저장된다.

문제 43. 정답: ② **| 해설:** [ㄱ] z = x%y = 7%2 = 1 (O) [ㄴ] x//y = 7//2 = 3 (O) [ㄷ] result = (1==1) and (3==3) ➜ True and True ➜ True (X)

문제 44. 정답: ④ **| 해설:** [4] result 변수에는 불리언 자료형인 True가 있으므로 출력하면 True가 출력된다. (O) [5] 'True'는 따옴표로 감싸져 있으므로 문자열형이다.

문제 45. 정답: ② **| 해설:** //는 몫을 구하는 연산자이다.

문제 46. 정답: 정수형과 문자열형은 더하기할 수 없으므로 오류가 발생한다. **| 해설:** 이 프로그램을 실행하면 다음과 같은 오류가 발생한다:

```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

3차시 표준 입력과 출력 [개념편 12쪽 / 문제편 22쪽]

문제 47. 정답: ○ **| 해설:** print() 함수는 파이썬에서 다양한 값들을 출력할 수 있는 기본 함수이다.

문제 48. 정답: X **| 해설:** input() 함수는 항상 입력받은 내용을 문자열 형식으로 되돌려준다. 따라서, 숫자 연산을 위해서는 int() 또는 float() 함수로 자료형을 정수 또는 실수로 변환해주어야 한다.

문제 49. 정답: X **| 해설:** 문자열과 정수는 더하기 연산을 할 수 없다. 연결된 하나의 문장을 만들기 위해서는 정수 1을 str(1)로 문자열 '1'로 변환해야 한다.

문제 50. 정답: X **| 해설:** int() 함수는 문자열이나 실수 등 다른 자료형의 데이터를 정수형(int)로 변환할 때 사용된다. 문자열로 바꾸려면 str() 함수를 사용해야 한다.

문제 51. 정답: ○ **| 해설:** f"문자열{변수}" 형식은 f-string으로, 저자가 개인적으로 생각하기에 가장 간편하고 직관적인 문자열 합치기 방법이다.

문제 52. 정답: ○ **| 해설:** .format() 메서드는 중괄호 {}에 순서대로 값을 삽입한다.

문제 53. 정답: ○ **| 해설:** print() 함수 내에서 콤마(,)로 구분한 항목 사이에는 기본적으로 공백 한 칸이 포함되어 출력된다. 참고로, 더하기 연산자로 합쳤을 때는 공백이 포함되지 않는다.

문제 54. 정답: ⑤ **| 해설:** [5] input()로 입력받은 값이 정수로만 이루어져 있다고 해서 자동으로 정수형 데이터로 값이 반환되지는 않는다. (X) [1] 파이썬 코드에서 # 기호 뒤에 글자를 적는 것을 주석이라고 하며 코드 중간에 메모를 적기 위한 용도로 사용한다. 코드 실행에는 영향을 주지 않는다. (O) [2] input()의 반환값은 항상 문자열형으로만 리턴된다. (O) [3] input 함수에 글자를 적으면, 그 인수는 콘솔 창에 입력하는 부분 왼쪽 설명 부분으로 표시된다.(개념편 12쪽) (O) [4] =는 할당 연산자라고 하며, 변수에 값을 저장할 때 사용한다. (O)

문제 55. 정답: ③ **| 해설:** [3] input() 함수는 항상 문자열로 리턴하지만, 코드에 int() 등을 사용하여 다른 자료형으로 변환할 수 있다. (O) [1] input() 함수는 항상 문자열로 입력받은 값을 리턴한다. (X) [2] 입력 안내 문구는 input() 함수 괄호 안에 작성한다. (X) [4] 반환값을 변수에 저장하지 않아도 실행은 되지만 값은 저장되지 않기에 사라진다. (X) [5] input() 함수로 입력받을 때 숫자, 문자열을 포함하여 뭐든지 입력받을 수 있다. (X)

문제 56. 정답: ① **| 해설:** 입력받은 "5"를 int()로 정수형으로 변환한 뒤 2배하였으므로 10이 출력된다.

문제 57. 정답: ② **| 해설:** [2] input()은 항상 입력받은 값을 문자열으로 반환하므로, str()은 생략해도 된다. (O) [1] int()를 생략하면 in1은 문자열이 되므로 기존과는 다른 동작을 한다. (X) [3]&[4] 정수와 문자열은 서로 더하기할 수 없으므로 오류가 발생한다. (X) [5] 문자열에서 큰따옴표/작은따옴표 종류는 관계없다. 단, 서로 같은 따옴표로 문자를 감싸야 한다. (X)

문제 58. 정답: ③ **| 해설:** [ㄷ] input() 함수는 항상 문자열을 반환하므로, 산술 연산을 위해서는 int()나 float() 같은 (명시적인) 자료형 변환이 필요하다. (O) [ㄱ] 따옴표가 포함된 값이 입력되면, "가 \"로 바뀐 상태로 불러와진다. 입력 자체가 문자열로 받아지며, 오류는 발생하지 않는다. (X) [ㄴ] input() 함수는 항상 입력받은 값을 문자열로 리턴하기 때문에 123을 입력해도 문자열 "123"로 저장되며, 파이썬은 자동으로 정수로 판단하지 않는다. (X)

문제 59. 정답: ⑤ **| 해설:** input() 함수는 항상 문자열을 반환하므로 정수 연산을 하기 위해서는 int() 함수로 자료형 변환이 필요하고, 괄호 안에 아무런 인수를 주지 않으면 아무 안내 메시지도 안 나온다.

문제 60. 정답: ④ **| 해설:** ㄱ input() 함수는 항상 입력받은 값을 문자열로 리턴한다. 정수로 만드려면 int()로 자료형을 변환해줘야 한다. (X) ㄴ print("a", 1, "b")에서 쉼표로 구분된 값 사이에는 기본적으로 자동으로 공백이 삽입된다. 더하기 연산자로 했을 때는 삽입되지 않는다. (O) ㄷ 문자열과 숫자는 + 연산이 불가능하다. (X) ㄹ f-string은 표현식을 계산하여 문자열에 삽입한다. 1+2 = 3이므로 "3입니다"가 출력된다. (O)

문제 61. 정답: ② **| 해설:** 문자열형과 정수형은 서로 더하기할 수 없으므로, 여러 개를 합쳐 출력하기 위해서는 정수형 데이터를 문자열형으로 변환해주어야 한다. 5 따옴표가 없어서 오류가 발생한다. (X)

문제 62. 정답: ④ **| 해설:** 문자열을 정수로 바꿔 연산하면 숫자 더하기가 가능하다.

문제 63. 정답: ② **| 해설:** format 메서드로 값이 삽입될 위치를 중괄호로 정확하게 지정할 수 있다.

문제 64. 정답: ⓐ x,"입니다." ⓑ {y} ⓒ 20 **| 해설:** 빈칸 a 앞에 쉼표가 있는 것으로 보아, 값들을 쉼표로 연결해야 했으며, 빈칸 b의 경우 따옴표 앞에 f가 있는 것으로 보아, f-string으로 변수와 문자열을 합친 것이다. 이때, 변수나 수식을 문자 사이에 넣으려면 중괄호를 사용한다. 빈칸 c는 입력받은 값을 바로 출력하는 것이므로 20이다.

문제 65. 정답: ⓐ f"입력한 단어는 {a}와 {b}입니다." ⓑ (a+b)*c **| 해설:** 문자열끼리 더하면 문자가 합쳐져 연결이 되며, 곱하면 그 횟수만큼 반복이 된다. 빈칸 b에 들어가는 표현식의 의미는 a와 b의 문자를 연결한 것을 c번 반복하라는 것이다.

문제 66. 정답: ⑤ **| 해설:** \n는 줄바꿈을 의미하는 이스케이프 시퀀스 기호이다.

문제 67. 정답: 7777 **| 해설:** 문자열을 곱하면 그 횟수만큼 반복한다. input() 함수의 반환값은 기본적으로 문자열형이다. 숫자 연산을 원한다면 int() 등으로 숫자로 변환해주어야 한다.

문제 68. 정답: {int(num)**2} **| 해설:** f-문자열 형식에서 변수나 수식을 문자 사이에 넣으려면 중괄호를 사용한다. 이때, num 변수는 사용자로부터 입력받은 값을 저장하는 변수로, input() 함수에게서 값을 받기 때문에 문자열의 데이터를 받는다. 따라서, 제곱된 값을 표현하려면 int(num)으로 정수형으로 자료형 변환을 한 상태에서 제곱을 해야 한다.

문제 69. 정답:

```
width = int(input("직사각형의 가로 길이:"))
height = int(input("직사각형의 세로 길이:"))
print("넓이:", a*b)
```

해설: 참고로, int() 함수를 input() 함수에 바로 감싸지 않고, 계산하는 부분에 변수명을 감싸서 코드를 작성해도 된다. 예를 들면, `a = input("가로 길이:")`를 하고, `int(a) * int(b)`를 하는 것이다.
또, 마지막 출력 부분에서 `print("넓이:", a*b)`도 가능하지만 `print(f"넓이: {a*b}")` 등으로 해도 된다.

항목		기준	배점
1	입력	input() 함수로 사용자로부터 입력을 받을 수 있는가?	1
2	출력	print() 함수와 문자열 합치기 기법으로 출력을 할 수 있는가?	2
3	자료형 변환	산술 연산 전에, 또는 산술 연산 중에 int() 또는 float()로 입력받은 값을 숫자 형식으로 형 변환할 수 있는가?	2

문제 70. 정답: ⓐ total//60 ⓑ total%60 **| 해설:** 60초가 1분과 같으므로, 전체 초를 60으로 나눈 몫은 분이 되고, 그 나머지는 초가 된다.

문제 71. 정답: ⓐ total//3600 ⓑ (total%3600)//60 ⓒ total%60 **| 해설:** 시간의 경우, 1시간이 3600초이므로 전체 초를 3600으로 나눈 값이고 분의 경우, 남은 초 중에서 60으로 나눈 몫이 분이므로 시간 값을 구하고 남은 초를 60으로 나눈 값, 초의 경우, 계산 후 남은 초이다.

문제 72. 정답: age >= 18 **| 해설:** >=는 왼쪽이 오른쪽보다 크거나 같은지를 확인하는 비교 연산자로, 참 또는 거짓을 알려준다. (개념편 11쪽)

문제 73. 정답: ④ **| 해설:** "user_input"은 문자열로 인식되기 때문에, 실제 변수 값을 쓰기 위해서는 user_input처럼 따옴표 없이 써야 한다.

문제 74. 정답: ①, ②, ④ **| 해설:** 1 input() 함수는 표준 입력 함수로, 사용자가 키보드로 입력한 값을 프로그램에 전달한다. 이때 입력되는 모든 값은 자료형과 관계없이 항상 문자열형으로 처리되며, 숫자를 입력하더라도 내부적으로는 문자열로 저장된다. (O)
2 사용자가 19를 입력해도 변수 age에는 "19"라는 문자열이 저장된다. 따라서 숫자처럼 보이더라도 실제로는 문자열이므로 숫자 형식(정수형 또는 실수형)으로 형 변환을 해야 산술 연산이 가능하다. (O)
4 + 연산자는 피연산자의 자료형에 따라 동작이 달라진다. 두 문자열 사이에서는 문자열을 이어주는 연결 연산자로, 두 숫자 사이에서는 산술 덧셈 연산자로 작동한다. (O)
3 int(age) + 1은 age를 정수형으로 바꾸는 코드이므로 이 연산은 문자열 연결이 아닌 정수 덧셈이다. "19"를 int()로 변환하여 19 + 1 = 20을 계산하는 것이므로 설명은 틀렸다. (X)
5 age + 1에서 age는 문자열 "19"이고 1은 정수이므로, 문자열과 숫자는 덧셈이 불가능하다. 파이썬은 서로 다른 자료형 간 덧셈을 허용하지 않기 때문에 TypeError가 발생한다. 따라서 오류 없이 문자열처럼 붙지 않는다.

문제 75. 정답: ② **| 해설:** "안녕하세요, 동규님!"이라는 출력이 나온 것으로 보아, 첫 번째 입력값으로 동규가 사용되었음을 알 수 있다. "내년에는 20 살이 되시겠네요!"라는 문장은 int(age) + 1의 결과이므로, age에 저장된 값은 19였음을 알 수 있다. 따라서 입력값으로 이름은 동규, 나이는 19가 정확하다.

문제 76. 정답: ⓐ adult * ad_num + child * ch_num ⓑ 3 **| 해설:** adult * 2 + child * 3 = 25000*2 + 16000*3 = 50000 + 48000 = 98000이므로 아동 수는 3이다.

문제 77. 정답: ②, ③, ④ **| 해설:** 2 lucky는 input()으로 받은 문자열이고 float(lucky)는 실수형으로 형변환하는 것이다. (O) 3 \n은 줄바꿈, \t는 탭 문자를 의미하므로 출력된 문자열에는 줄바꿈과 탭이 모두 포함되어 있다. (O) 4 문자열과 정수는 + 연산자로 연결할 수 없기 때문에 str(age_num)으로 자료형을 변환하지 않으면 오류가 발행한다. (O) 1 17은 input()의 특징으로 인해 문자열형이지만 숫자 형식의 문자열이므로 float("17")은 17.0으로 잘 변환된다. (X) 5 print() 함수는 콤마(,)로 구분하면 문자열과 숫자를 함께 출력할 수 있다. (X)

문제 78. 정답: ① **| 해설:** 사용자가 input() 함수를 통해 값을 입력할 때, 문자 그대로 입력하는 것이지 프로그래밍 코드처럼 따옴표를 붙여 입력할 필요 없다. 사용자가 서영이라고 입력하면, 파이썬은 내부적으로 이를 "서영"이라는 문자열로 자동 처리되는 반면에, 사용자가 "서영"이라고 입력하면, 문자열 안에 따옴표까지 포함된 값이 되어 "\"서영\""이 저장되므로, 원하는 값과 달라질 수 있다. (문제편 13쪽)

문제 79. 정답: ③ **| 해설:** "삼점일사"는 숫자가 아니므로 float("삼점일사")에서 ValueError가 발생한다.

문제 80. 정답: ③ **| 해설:** 3 input() 함수는 사용자가 입력한 값을 항상 문자열형으로 저장한다. (O)
1 & 5 input()는 문자열형으로 저장하므로 숫자를 원할 경우 int() 등으로 형변환해야 한다. (X) 2 print() 함수는 여러 데이터를 콤마(쉼표)로 나열해서 출력할 수 있으며, 더하기 연산자는 문자열끼리만 연결할 때 사용된다. (X) 4 print()는 문자열, 변수, 숫자 등 모든 자료형을 출력할 수 있다. (X)

4차시

조건문과 흐름 제어 [개념편 15쪽 / 문제편 34쪽]

문제 81. 정답: ○ **| 해설:** 파이썬은 들여쓰기를 이용해 코드 블록을 정의한다. 일반적으로 공백 4칸을 사용한다.

문제 82. 정답: X **| 해설:** if문은 조건이 참일 때만 블록 안의 코드가 실행된다.

문제 83. 정답: ○ **| 해설:** 파이썬의 조건문에서 if, elif, else 뒤에는 반드시 콜론이 필요하다.

문제 84. 정답: ○ **| 해설:** 중첩 조건문은 조건문 내부에 또 다른 조건문이 들어있는 구조이다.

문제 85. 정답: X **| 해설:** else 블록에 들어있는 코드는 조건이 거짓일 때 실행된다.

문제 86. 정답: X **| 해설:** elif는 파이썬에서 정의된 예약어이지만, else if는 파이썬 문법에 존재하지 않는다. 반드시 elif라고 적어야 한다.

문제 87. 정답: X **| 해설:** 파이썬에서 소수를 0과 1로 이루어진 이진수로 근사값으로 저장되기 때문에 계산 과정에서 미세한 오차가 생겨 항상 True가 되지 않는다. 소수 계산과 비교는 신뢰할 수 없다.

문제 88. 정답: ○ **| 해설:** 비교 연산의 결과는 참(True) 또는 거짓(False)으로 된 불리언형이다.

문제 89. 정답: ○ **| 해설:** 논리 연산자 or는 둘 중 하나라도 참이면 결과가 참이다. 둘 다 거짓일 때만 거짓이 된다.

문제 90. 정답: X **| 해설:** 논리 연산자는 조건문 외에도 다양한 상황에서 사용될 수 있다. (예: 72번)

문제 91. 정답: ○ **| 해설:** if문은 조건이 참일 경우 블록 내 여러 줄의 코드가 모두 실행된다.

문제 92. 정답: X **| 해설:** 조건문 블록 안에는 어떤 파이썬 코드도 올 수 있으며, print()도 포함된다.

문제 93. 정답: ① **| 해설:** 순차 구조는 조건 없이 위에서 아래로 순서대로 실행되는 구조이다.

문제 94. 정답: ④ **| 해설:** 반복 구조는 반복 중단이 아니라 반복 실행을 위한 구조이다. 뒷 차시에서 다루는 내용이지만, break 키워드를 사용하면 반복을 멈출 수 있다.

문제 95. 정답: ④ **| 해설:** 선택 구조는 어떠한 조건을 기준으로 실행할 코드를 선택하여 실행하는 구조이다.

문제 96. 정답: ③ **| 해설:** [3] 중첩 조건문은 조건문 안에 또 다른 조건문이 있는 구조이다. (○) [1] 조건문 안에 반복문이 있는 것은 중첩 조건문이 아니라, 조건문과 반복문을 함께 사용하는 구조일 뿐이다. (X) [2] 여러 개의 조건을 and나 or로 연결하는 구조는 복합 조건문이라고 부른다. (X) [4] 조건 없이 무조건 실행되는 것은 순차 구조이다. (X) [5] 조건문 없이 실행되는 것은 일반적인 코드 블록이며, 조건문과 직접적인 관련이 없다. (X)

문제 97. 정답: ② **| 해설:** score가 85이므로 score >= 80 조건이 참이 되어 "B등급"이 출력된다. 첫 번째 조건은 거짓이므로 무시된다.

문제 98. 정답: ③ **| 해설:** [ㄱ] if문 뿐만 아니라 if, elif, else, for, while 등의 제어문 뒤에는 반드시 콜론(:)이 필요하며, 없으면 문법 오류가 발생한다. (○) [ㄴ] 파이썬은 {} 대신 들여쓰기로 코드의 블록(범위)을 구분한다. (○) [ㄷ] elif는 if 없이 단독으로 사용할 수 없다. 반드시 if 다음에 와야 한다. (X)

문제 99. 정답: ④ **| 해설:** ㄱ else는 생략할 수 있다. (O) ㄷ pass는 코드는 필요하지만, 아무 작업도 하지 않기를 원할 때 사용하는 것이다. (개념편 17쪽 상단) (O) ㄴ elif와 else는 순서를 지켜야 한다. if, elif, elif, ..., else 순서이다. (X)

문제 100. 정답: ④ **| 해설:** =는 값을 저장할 때 사용하는 할당 연산자이며, 같음을 비교할 때는 반드시 ==를 사용해야 한다.

문제 101. 정답: ④ **| 해설:** and, or, not은 파이썬에서 논리 연산자로 사용되며, 여러 조건을 한 줄에 연결하여 복합 조건식을 표현할 수 있게 해준다. ①,②는 서로 반대로 설명되어 있어 틀렸고, ③,⑤는 의미가 틀렸다.

문제 102. 정답: ③ **| 해설:** ㄴ `if x:`의 블록은 x의 자료형이 정수형, 문자열형, 어떤 것이든 참처럼 평가되는 값이면 실행된다. 예를 들어 x가 정수형 5일 때, `if x:`는 참이다. 0이 아니기 때문이다. 이때, **숫자 0은 False, 1은 True로 간주되고 문자열의 경우 비어있는 문자열은 False, 내용이 있는 문자열은 True로 간주**된다. (O) ㄷ not 0은 True이다. 0은 거짓으로 간주되고, 그 부정은 참이므로 블록이 실행된다. not False는 True이기 때문이다. (O) ㄱ elif는 앞선 조건이 모두 거짓일 때만 검사되며, 참이라도 앞에서 이미 참이 있었다면 실행되지 않는다. (X)

문제 103. 정답: ④ **| 해설:** x의 값이 즉 50 이상 60 이하인 경우, 0인 경우, 100인 경우에 참이 되는 조건문이다. 이때, 70은 0, 100이 아니고 50 이상 60 이하인 수가 아니므로 참이 되지 않는다.

문제 104. 정답: ⓐ `x % 2 != y % 2` ⓑ `x % 2 == 0` ⓒ `x >= y` **| 해설:** 첫 번째 조건 `if (x % 2 == 0 and y % 2 != 0) and` 빈칸 ⓐ`:`에서 (6,3)일 때 조건 1이 실행되고, (4,4)일 때는 다음 조건으로 넘어간 것을 보면, ⓐ가 (6,3)에서는 True, (4,4)에서는 False여야 한다. 즉, x와 y가 짝수/홀수로 다를 때만 첫 번째 조건이 만족되어야 하므로, `x % 2 != y % 2`가 적절한 조건이다. 두 번째 조건 `elif (x > y and` 빈칸 ⓑ `) or (x == y and x % 2 == 0):` 에서 (8,5)는 x > y이면서 x가 짝수이므로, ⓑ는 `x % 2 == 0`이 되어야 두 번째 조건이 참이 된다. 또한 (4,4)는 x == y이고 짝수이므로 역시 조건 2를 만족하지만, (5,5)는 홀수라서 두 번째 조건을 만족하지 않는다는 점에서 `x % 2 == 0`이 적절하다. **|** 세 번째 조건 `elif not` 빈칸 ⓒ`:` 은 (3,5)일 때 실행된다. 이 경우, 앞의 조건들이 모두 거짓이고, x < y이므로 x >= y가 거짓이다. 따라서, not (x >= y)는 참이 되어 ⓒ는 x >= y가 적절하다. **시험에 이런 문제가 나오면 넘어갔다가 뒤에꺼까지 풀고 다시 온다!**

문제 105. 정답: A **| 해설:** if 조건이 참이면 그 다음에 있는 elif 조건은 무시된다.

문제 106. 정답: 거짓 **| 해설:** 수학에서 `2<3<4`를 썼을 때 연결지어 동시에 확인하는 것처럼, 파이썬은 `(2<3) and (3<4)`으로 이해한다. 마찬가지로, 파이썬은 `a < b == True`를 `(a < b) and (b == True)`으로 이해한다. 그래서 두 조건이 모두 참이어야 참이 된다.

문제 107. 정답: PASS **| 해설:** and가 or보다 우선순위가 높아, x == 2 or (x == 3 and y == 3)로 해석된다. x == 2 ➜ True ➜ 전체 조건 True ➜ PASS

문제 108. 정답: ③ **| 해설:** x = 7, y = 4 등 입력값을 임의로 두고 직접 조건문을 풀어보면 된다.

문제 109. 정답: B **| 해설:** 주어진 변수는 a = 6, b = 3이다. 먼저 조건문에서 if a % 2 == 1을 검사하면, 6 % 2 = 0이므로 조건은 거짓이 되어 해당 블록은 실행되지 않는다. 다음으로 elif a > b and b % 3 == 0 조건을 확인하면, 6 > 3은 참이고, 3 % 3 == 0도 참이므로 두 조건이 모두 만족되어 "B"가 출력된다. 이때 elif 조건이 만족되었기 때문에 이후 조건인 elif b > 0은 검사하지 않고 넘어가며, 프로그램은 "B"만 출력하고 종료된다.

문제 110. 정답: ② **| 해설:** 조건연산자(삼항연산자)는 `참일 때 값 if 조건 else 거짓일 때 값` 구조다.

5차시 반복문과 흐름 제어 [개념편 18쪽 / 문제편 43쪽]

문제 111. 정답: X **| 해설:** range(3)는 0부터 시작하여 0, 1, 2 (총 3개)를 생성한다.

문제 112. 정답: O **| 해설:** while는 조건이 True인 동안 계속 반복되므로 무한 반복이다.

문제 113. 정답: O **| 해설:** <=이므로 5도 포함되어 반복된다.

문제 114. 정답: O **| 해설:** range(5)는 0~4까지 숫자를 포함하며, list()로 변환하면 리스트가 된다.

문제 115. 정답: O **| 해설:** 조건 기반 반복이므로 횟수 기반 for문과 다르게 유동적인 반복에 적합하다.

문제 116. 정답: X **| 해설:** break는 반복문을 아예 종료시킨다. 일시 정지가 아니라 완전 종료이다.

문제 117. 정답: O **| 해설:** while에 붙은 else는 반복 조건이 거짓이 되었을 때 한 번만 실행된다.

문제 118. 정답: O **| 해설:** continue로 인해 `i`가 4일 때는 다음 반복으로 건너뛴다.

문제 119. 정답: ⑤ **| 해설:** [5] 코드에서 data의 위치가 range()가 들어가는 위치가 맞지만, range() 만의 위치가 아닌, 반복되는 모든 객체가 들어갈 수 있는 자리이다. 리스트가 들어가도 되고, 문자열이 들어가도 되고, range()도 들어갈 수 있다. 여기서 data 변수는 리스트를 가지고 있으므로 리스트라고 해야 한다.

문제 120. 정답: ② **| 해설:** 1 + 2 + 3 + 4 = 10이므로 n=4일 때 출력이 10이다.

문제 121. 정답: ③ **| 해설:** [ㄱ] for n in data:는 리스트 요소를 하나씩 꺼내와 n에 저장하며 반복한다. 이때, n % 2 == 0에서 n이 짝수인 경우만 result에 더하므로 짝수 요소의 합을 구한다. (O) [ㄴ] 조건을 만족하는 값 n 자체를 더해야 하므로 result += n이 되어야 한다. (O) [ㄷ] 짝수는 2 하나뿐이므로 결과는 2이다. (X)

문제 122. 정답: ④ **| 해설:** [ㄱ] range() 함수는 처음과 끝을 주어줬을 때 끝-1의 숫자까지만 생성하므로 7까지가 아니라 6까지를 포함한다. (X)

문제 123. 정답: ③ **| 해설:** range(3, 0, -1)은 3에서 시작해서 1까지 감소한다는 의미이다.(마지막 값은 포함하지 않기 때문) 따라서 정답은 [3, 2, 1]이다.

문제 124. 정답: ① **| 해설:** range(1, 5)는 1, 2, 3, 4까지 반복한다는 것이므로 1+2+3+4=10이다.

문제 125. 정답: ③ **| 해설:** 현재 for문은 num의 각 값이 3 초과일 때 total에 더하는 것인데, 리스트 nums에서 n>3인 값은 4와 6이므로 초기 total의 값 0에 +4, +6이 된다. 따라서, 출력되는 값은 10이다.

문제 126. 정답: 012 **| 해설:** range(3)은 0부터 시작하여 3 이전까지의 정수인 0, 1, 2가 나오고 마지막 print() 함수에는 end 매개변수로 출력 뒤에 아무 글자도 넣지 않는다(\n 줄바꿈도 없음)고 적어놨으므로 결과는 012이다. 만약, end 매개변수가 주어지지 않았다면 0, 1, 2로 총 3줄에 걸친 결과가 나왔을 것이다.

문제 127. 정답: [(가)] continue [(나)] b [(다)] b가 10 이하입니다. **| 해설:** 이 프로그램은 사용자가 입력한 정수 a가 짝수인지 홀수인지에 따라 다른 방식으로 b에 값을 누적한 뒤, 그 결과를 출력하고 조건에 따라 메시지를 보여주는 구조이다. 만약에, 입력된 값이 짝수라면 1부터 a까지의 모든 정수를 더해 b에 저장한다. 입력된 값이 홀수일 경우에는 0부터 a-1까지 순회하면서, 짝수인 값은 건너뛰고 홀수인 값만

b에 더한다. 그렇기에 사용자가 입력창에 5를 입력했을 때, i는 0부터 4까지 반복되며, i % 2 == 0인 경우 continue를 통해 반복을 건너뛰므로 1과 3만 더해져 b = 4가 된다. 이후 print(b)가 실행되어 4가 출력되며, 이어지는 조건문에서는 b가 10보다 작기 때문에 "b가 10 이하입니다."가 출력된다.

문제 128. 정답: ⓐ nums ⓑ n % 2 == 1 ⓒ n **| 해설:** nums 리스트의 홀수 합을 구하는 것이므로 빈칸 ⓐ에는 변수 이름인 nums가 들어가야 하고, 짝수는 2로 나누었을 때 나머지가 0이고 홀수는 1이므로 빈칸 ⓑ는 n % 2 == 1가 되어야 한다. 마지막으로 빈칸 ⓒ는 값을 더해야 하는 것이므로 n이 들어가야 한다.

문제 129. 정답: ③ **| 해설:** 변수 result는 1부터 n까지의 곱을 계속 저장하는 역할을 하며, 곱셈 결과를 누적으로 저장한다.

문제 130. 정답: ② **| 해설:** 빈칸에는 `range(1, n + 1)`가 들어가야 하며, 1부터 n까지 포함된다는 의미이므로 1부터 n까지 반복한다.

문제 131. 정답: ③ **| 해설:** 6! = 6×5×4×3×2×1 = 720 이다.

문제 132. 정답: ④ **| 해설:**

ㄴ i=1, j=1 ➜ 1+1=2 [✔]
i=1, j=2 ➜ 1+2=3 [✔]
i=2, j=1 ➜ 2+1=3 [✔]
i=2, j=2 ➜ 2+2=4 [✔]
i=3, j=1 ➜ 3+1=4 [✔]
i=3, j=2 ➜ 3+2=5 [✘] 이므로 총 5회 실행된다. (O)

ㄷ 1번째 줄에서 변수 total의 초기값은 0이다. (O) ㄱ 2+3+3+4+4=16이다. (X)

문제 133. 정답: ⑤ **| 해설:** 반복 횟수를 알고 있으니 충분히 for문으로 바꾸더라도 같은 결과를 낼 수 있다.

```
total = 0
for n in range(1, 6):
    total += n
print("합계:", total)
```

문제 134. 정답: ③ **| 해설:** 3 while문은 조건이 참인 동안 반복이 실행된다. 따라서, while True: 는 무한 반복 루프이다. (O) 1 while문은 조건이 참인 동안 반복이 실행된다. (X) 2 while문의 조건, 블록의 코드에 따라 반복 횟수는 달라진다. (X) 4 else는 선택사항이며, else가 있다면 while문의 조건이 거짓일 때 한 번만 실행된다. (X) 5 while문을 리스트와 함께 사용하는 것은 의무 아니다. (X)

문제 135. 정답: ① **| 해설:** while문은 조건이 참인 동안 무한 반복되므로 i값에 따라 반복이 될 수도 있고 안될 수도 있는 3번과 4번 선택지는 안 된다. 또, 2번 선택지와 5번 선택지는 조건이 항상 거짓이므로 안 된다. 숫자 0은 False로 간주되고, 숫자 1은 True로 간주된다. 따라서, 항상 무한 반복이라 할 수 있는 것은 while True: 와 while 1: 이다.

문제 136. 정답:

```
1
2
3
4
5
끝
```

| 해설: while 조건이 False가 되면 else 블록이 1번 실행된다.

문제 137. 정답: ① **| 해설:** 바깥쪽 while문에서 **i**는 1부터 시작하여 3까지 반복된다. 안쪽 while은 **j**가 1부터 시작하여 2까지 반복된다. 실행하였을 때의 **i**와 **j**의 값 변화를 보도록 하자.

i=1 일 때, **j=1** ➜ 1-1| 출력 **j=2** ➜ 1-2| 출력 **j=3** ➜ j<=2가 False ➜ 종료 **i+=1** ➜ **i=2**	**i=2** 일 때, **j=1** ➜ 2-1| 출력 **j=2** ➜ 2-2| 출력 **j=3** ➜ j<=2가 False ➜ 종료 **i+=1** ➜ **i=3**	**i=3** 일 때, **j=1** ➜ 3-1| 출력 **j=2** ➜ 3-2| 출력 **j=3** ➜ j<=2가 False ➜ 종료 **i+=1** ➜ **i=4**

문제 138. 정답: ④ **| 해설:** [4] 조건이 항상 참(True)이라면 while문은 끝나지 않고 무한 반복될 수 있다. (O) [1] while문은 조건이 참인 동안 계속 반복되므로 횟수를 미리 지정할 필요는 없다. (X) [2] break는 반복을 종료하며, 건너뛰는 것은 continue이다. (X) [3] continue는 반복을 종료하는 것이 아니라 블록의 아래 코드를 무시한 후 다음 반복으로 건너뛴다. (X) [5] continue는 반복문 내부에서만 사용해야 하며, 바깥에서는 오류가 발생한다. (X)

문제 139. 정답: ② **| 해설:** [ㄱ] i==3일 때 continue가 실행되어 아래 print(i)는 건너뛰고 다음 반복으로 넘어간다. (O) [ㄴ] i==5일 때 break가 실행되어 반복문이 종료된다 (O) [ㄷ] 3은 continue로 인해 출력되지 않기 때문에 결과에 포함되지 않는다. (X)

문제 140. 정답: ④ **| 해설:** i는 1부터 3까지 while 반복 (총 3회) j는 1부터 3까지 for 반복 (총 3회)

i	j	i==j?	continue 아래 코드 실행	i+j	total
1	1	O	X		0
1	2	X	O	3	3
1	3	X	O	4	7
2	1	X	O	3	10
2	2	O	X		10
2	3	X	O	5	15
3	1	X	O	4	19
3	2	X	O	5	24
3	3	O	X		24

문제 141. 정답: ⓐ 6 ⓑ 1 **| 해설:** i=1부터 시작해서 i<6일 동안 반복되고 짝수일 때만 total+=i*2가 되며 매번 i는 1만큼 증가하였을 때 i=2 ➜ 2*2=4 / i=4 ➜ 4*2=8 // total =4+8=12이다.

문제 142. 정답: 18 **| 해설:** 매 반복마다 a += 2이다. 즉, a는 1➜3➜5까지 변화

a = 1일 때, 1은 홀수 **total += 1 * 2 ➜ 2** **a += 2 → a = 3**	**a = 3**일 때, 3은 홀수 **total += 3*2 ➜ 2+6 ➜ 18** **a += 2 → a = 5**	**a = 5**일 때, 5는 홀수 **total += 5*2 ➜ 8+10 ➜ 18** **a += 2 → a = 7 (while** 종료)

문제 143. 정답: ② **| 해설:** while True: 반복이므로 "3" 입력 전까지 반복된다.

문제 144. 정답: ⑤ **| 해설:** "첫 번째 숫자: 10"은 입력 프롬프트로 출력되는 것이며, 프로그램의 출력 결과는 아니다.

문제 145. 정답: ② **| 해설:** choice == "1"일 때는 x + y ➜ 7 + 4 = 11 이다.

문제 146. 정답: ⓐ "*" ⓑ dan*i **| 해설:** 출력은 <단> * <i> = <단과 i를 곱한 값>의 형태이므로 첫 번째 빈칸에는 곱하기 기호, 두 번째 빈칸에는 곱한 결과가 들어가야 한다.

6차시 파일과 예외 상황 제어 [개념편 21쪽 / 문제편 60쪽]

문제 147. 정답: X **| 해설:** "w" 모드는 기존 내용을 삭제하고 새로 쓰기에 기존 내용은 유지되지 않는다.

문제 148. 정답: X **| 해설:** "w" 모드는 파일이 없으면 새로 생성한다. 단, 폴더 자체가 없으면 오류

문제 149. 정답: O **| 해설:** read() 메서드는 전체 파일 내용을 문자열로 반환한다.

문제 150. 정답: X **| 해설:** with-as 블록을 벗어나면(with문 밖으로 빠져나갔을 때) 파일이 자동으로 닫혀 객체는 무효가 된다.

문제 151. 정답: X **| 해설:** "w"는 쓰기 전용 모드로, 여기서 읽기를 시도하면 오류가 발생한다.

문제 152. 정답: X **| 해설:** write()는 문자열만 가능하므로 숫자는 str()로 문자열로 형 변환해야 한다.

문제 153. 정답: ④ **| 해설:** [4] "r"은 일반 텍스트 읽기 모드로 가장 기본적인 읽기용이다. (O)
[1] "w"는 쓰기 모드이다. (X) [2] "a"는 추가 모드이다. (X)

문제 154. 정답: ④ **| 해설:** read()는 파일을 읽을 때 사용하며, 쓰기 목적과는 맞지 않다.

문제 155. 모범 답안: 서로 다른 자료형을 합칠 수 없다. ID라는 변수는 정수형 데이터가 들어있기 때문에 프로그램이 실행되면 문자열과 숫자를 합치게 된다. 따라서, str(ID)로 바꿔야 한다.

문제 156. 정답: ⓐ f.read() ⓑ f.close() **| 해설:** print(d)로 파일의 내용을 출력하므로 변수 d에는 f.read()으로 파일의 값을 읽어와야 하고, 항상 파일 작업 끝에는 파일을 닫아야 하므로 f.close()를 사용한다.

문제 157. 정답: ④ **| 해설:** [4] "a" 모드는 append 모드로, 기존 내용 뒤에 데이터를 추가한다. (O)
[5] with 구문을 사용했기 때문에 파일은 자동으로 닫힌다.

문제 158. 정답: ① **| 해설:** 파이썬의 파일 쓰기, 파일 닫기, 파일 읽기, 출력 흐름에 따라 작성된 코드로,
ㄴ: 파일을 "w"로 열기 (쓰기 준비)
ㄷ: "파이썬 파일입출력" 문자열을 파일에 씀
ㄹ: 파일 닫기 (쓰기 완료)
ㅁ: 파일을 "r"로 열기 (읽기 준비)
ㅂ: 파일의 내용을 읽어 data 변수에 저장
ㅅ: 파일 닫기
ㄱ: 읽은 내용을 화면에 출력

문제 159. 정답: ② **| 해설:** [1] 메서드는 객체 뒤에 점(.) 기호를 적은 다음 사용해야 한다. (X)
[3] 추가 모드에서는 읽기 작업을 할 수 없다. (X) [4] open()은 메서드로 사용하지 않는다. (X)
[5] 쓰기 모드에서는 읽기 작업을 할 수 없다. (X)

문제 160. 정답: ② **| 해설:** [2] "r" 모드는 읽기 모드이며, 파일이 없으면 새로 만들지 않고, 오류가 발생한다. (X) [1] open(파일 경로, 파일 모드) 형태로 사용한다. 경로와 모드는 따옴표로 감싸야 한다. (O) [3] "w" 모드는 쓰기 모드로, 파일이 없으면 새로 만들고, 저장할 때에는 기존 내용을 남기지 않고 덮어쓰기 한다. (X) [4] "a" 모드는 추가 모드로 파일의 내용 뒤에 새로운 내용을 추가한다. (X)
[5] with문은 자동으로 close()를 호출해 주는 안전한 방식이다. (X)

문제 161. 정답: ⑤ **| 해설:** [5] 윈도우에서 경로 구분자는 \이지만, 파이썬에서는 문자열에서 \가 \n, \t처럼 특수한 의미를 가지고 있으므로 그 자체를 문자로 사용하려면 \\로 두 번 써야 한다. (O) [1] \n, \t는 이스케이프 문자로 인식되어 자동 삭제되지 않고, 줄바꿈, 탭 등으로 처리된다. 경로 오류가 발생할 수 있다. (X) [2] open() 함수는 읽기 모드에 따라 파일은 자동 생성할 수 있지만, 존재하지 않는 폴더는 생성하지 않는다. (X) [3] demo.txt는 txt라는 확장자가 존재한다. 확장자 없이 demo만 적으면 오류가 발생한다. (X) [4] 파이썬에서는 D드라이브, E드라이브, 또는 상대 경로 등 다양한 경로 방식이 가능하다. (X)

문제 162. 정답: ④ **| 해설:** [4] write()는 데이터를 메모리라는 곳에 임시로 기록하며, 파일을 닫지 않으면 실제 파일에 반영되지 않을 수 있다. (X) [1] "a" 모드는 추가 모드로, 기존 파일 내용 뒤에 내용을 덧붙이는 방식이다. 기존 데이터는 지워지지 않는다. (O) [2] input()은 콘솔에서 사용자로부터 문자열을 입력받는 내장 함수이다. (O) [3] \n을 앞에 붙여야 줄바꿈 후에 입력값이 추가되어 가독성이 좋아진다. (O) [5] 파일을 열고 쓰기를 한 뒤 close()를 호출해야 실제 파일에 저장된다. close()를 생략하면 내용이 파일에 저장되지 않거나, 파일 이동/삭제가 막힐 수 있다. (O)

문제 163. 정답: ① **| 해설:** read(3)은 처음부터 3글자인 P,y,t만 읽어 출력한다.

문제 164. 정답: ③ **| 해설:** for line in f:는 줄바꿈 기준으로 3번 반복하게 되어 총 3줄이 출력된다.

문제 165. 정답: ③ **| 해설:** readline()은 첫 번째 줄만 읽으며 줄 끝의 \n까지 포함하여 출력된다.

문제 166. 정답: ④ **| 해설:** except는 반드시 try와 함께 사용해야 하며, 단독으로 사용할 수 없다.

문제 167. 정답: ③ **| 해설:** finally 블록은 예외 발생 여부와 관계없이 반드시 실행된다. 따라서 어떤 상황에서도 자원 정리(파일 닫기 등)에 유용하다.

문제 168. 정답:

```
0으로 나눌 수 없습니다.
나눗셈 완료.
```

| 해설: divide(0)을 호출하면 10 / 0이 수행되고 ZeroDivisionError 예외가 발생한다. 따라서 try 블록의 print("결과:", result)는 실행되지 않고 바로 except ZeroDivisionError: 블록이 실행된다. 이후 print("나눗셈 완료.")는 try-except 바깥이므로 마지막에 항상 실행된다.

문제 169. 정답:

```
입력 받은 숫자: 42
b
c
```

| 해설: 사용자가 42를 입력했으므로 int("42")는 예외 없이 정상 작동하고, try 블록 안의 print("입력 받은 숫자:", num)가 정상적으로 실행된다. 이때, 예외가 없으므로 except는 건너뛴다. 이후, else 블록이 실행되어 "b"가 출력된다. 마지막으로, finally는 항상 실행되므로 "c"가 출력된다.

7차시 함수와 라이브러리 [개념편 26쪽 / 문제편 69쪽]

문제 170. 정답: ○ **| 해설:** 파이썬에서는 함수를 정의할 때 def 키워드를 사용해야 한다.

문제 171. 정답: ○ **| 해설:** print(), input(), max() 등은 파이썬에 내장된 함수들이다. 이들은 우리가 새로 정의해서 만들지 않아도 이미 정의되어 있는 상태이다. (개념편 26쪽 중앙)

문제 172. 정답: ○ **| 해설:** 함수를 정의할 때 기본값이 없는 매개변수(필수 인수)는 앞에, 기본값이 있는 매개변수(기본 인수)는 뒤에 와야 한다. 이를 지키지 않으면 오류가 발생한다.

✗	✓
`def greet(message="하이", name):`	`def greet(name, message="하이"):`

문제 173. 정답: ○ **| 해설:** 함수 안에서 전역변수를 수정하려면 global 키워드를 사용해야 한다.

문제 174. 정답: X **| 해설:** 함수 내부에서는 전역 변수보다 지역 변수가 우선되어 같은 이름의 변수가 함수 밖과 함수 안에 있다면, 함수 안에서는 지역 변수로 판단한다.

문제 175. 정답: X **| 해설:** 함수 안에서는 전역 변수보다 매개변수가 우선된다.

문제 176. 정답: ○ **| 해설:** 정의된 함수는 그 이후에 어디서든 호출할 수 있으며, 함수를 호출하는 시점에는 이미 함수가 정의되어 있는 상태여야 한다.

문제 177. 정답: ② **| 해설:** 파이썬의 변수와 함수 이름은 숫자로 시작할 수 없다.

문제 178. 정답: ② **| 해설:** 기본 인수는 반드시 오른쪽 끝부터 채워야 한다.

문제 179. 정답: ④ **| 해설:** [4] 위치 인수는 매개변수의 순서대로 인수를 함수에게 전달하는 방식이다. (○) [1]&[3] 함수를 호출할 때 매개변수의 정해진 순서대로 값을 넣지 않고, 어떤 값이 어떤 매개변수의 값을 의미하는지 이름을 써서 직접 지정해주는 방식은 키워드 인수 방식이다. (X)

문제 180. 정답: ④ **| 해설:** [4] 함수는 재사용성을 높이고, 코드 중복을 줄이는 데 도움이 된다. (○) [1] 매개변수 없이도 함수는 정의될 수 있으며, 매개변수의 개수에 대한 규칙은 존재하지 않는다. (X) [2] 파이썬은 사용자 지정 함수를 정의할 때 def 키워드를 사용한다. (X) [3] 파이썬은 인터프리터 언어로, 전체 코드를 한꺼번에 번역하고 실행하는 컴파일러 언어와 달리 한 줄씩 위에서 아래로 순차적으로 해석하고 실행한다. 따라서, **파이썬은 함수를 호출할 때 그 함수가 이미 완전히 정의되어 있어야 한다.** 반면에, 대표적인 컴파일러 언어인 C언어의 경우 함수를 나중에 정의할 수 있는 방법이 존재한다. (X) [5] 함수에는 반환값이 있어도 되고 없어도 된다. (X)

문제 181. 정답: ⑤ **| 해설:** [5] greet 함수는 두 개의 매개변수를 필요로 하지만, "민수" 하나만 전달되어 오류가 발생한다. (○) [1] age는 파이썬의 예약어가 아니다. (X) [2] 순서를 바꿔도 받은 인수의 개수가 부족하다. (X) [3] return이 없어도 함수 실행에 문제는 없다. (X)

문제 182. 정답: ④ **| 해설:** [ㄱ] global 키워드는 함수 내부에서 외부 변수를 수정하려고 할 때 반드시 필요하다. global은 전역 변수라는 선언만 할 수 있다. 'global 변수명 = 값'으로 값을 넣을 수는 없다. 반드시 따로 나눠야 한다. (○) [ㄷ] 코드의 while True: 무한 루프는 get_input()에서 리턴받은 반환값이 False일 경우 continue로 반복되고, True일 때 break로 종료된다. (○) [ㄴ] input()로 입력받은 값은 항상 기본적으로 문자열이므로 int()로 수치형으로 바꾸어주어야 한다. (X)

문제 183. 정답: ② **| 해설:** square(3)은 3**2=3^2=9를 반환하고, b는 문자열 "2"이므로 type(b)는 <class 'str'>이다.

문제 184. 정답: ① **| 해설:** 함수 내부에서 total이라는 변수에 값을 대입하지 않았기 때문에, 지역 변수는 없고, 지역 변수가 없으니 파이썬은 이 변수를 전역 변수로 인식한다. 따라서 이 <프로그램>에서는 global 키워드 없이도 값을 읽을 수 있다.

문제 185. 정답: 12 **| 해설:** 정수 1, 2를 문자열로 바꾼 후 결합한 것이다. 문자열끼리 더하면 붙이는(연결하는) 것이므로 12가 된다.

문제 186. 정답: 127 **| 해설:** mystery(5)에서 함수 mystery는 return x + x ** 3을 하므로 5 + 5 ** 3 = 5 + 125 = 130. 이때, 바깥 print 함수에서 -3을 하므로 127이다.

문제 187. 정답: ② **| 해설:** 기본 인수인 y=2를 사용하여 10 // 2 = 5가 되게 한다.

문제 188. 정답: ③ **| 해설:** [ㄱ] 두 번째 줄의 조건문에서, 두 수가 모두 양수여야만 num1**num2이 실행된다. (O) [ㄷ] process(2, 3)은 2**3=8, 즉, 8을 반환한다. (O) [ㄹ] 5번째 줄에 관련 코드가 존재한다. (O) [ㄴ] 만약 num1 또는 num2 중 하나가 0 이하이면 "음수 또는 0이 포함됨" 이라는 문자열이 반환되므로 이 경우에 result는 문자열이 된다. (X)

문제 189. 정답: 1 **| 해설:** x가 0이다. 조건문에서 0은 False로 간주되므로, x * 2는 실행되지 않고, return x + 1이 실행된다.

문제 190. 정답: 5번 줄 **| 해설:** if i = "a":는 대입문(=)을 사용한 것으로, 비교 연산자(==)를 사용해야 적절하다.

문제 191. 정답: ② **| 해설:** total(3)이 호출되면 double(3) + double(4)이 계산되어 리턴(반환)된다. double(3)은 3*2=6이고, double(4)는 4*2=8이다. 따라서, 6+8=14이므로 14라는 반환값이 리턴되기에 출력되는 값은 14이다.

문제 192. 모범 답안:

```
def add_num(a, b):
    return a+b
```

| 해설: 함수는 def 키워드로 정의할 수 있으며, "두" 수를 더한다 했으므로, 각 수를 받을 매개변수는 총 2개가 있어야 할 것이다. 또, 계산한 값을 리턴(반환)해야 하므로 return을 사용한다.

문제 193. 정답: 30 **| 해설:** 1부터 20까지의 숫자 중 짝수만 골라서 차례로 더하는 프로그램이다. 더한 값이 limit보다 크거나 같아지면 더하지 않고 멈춘다. 이때, 함수에 25를 넣었으므로 limit은 25가 된다. 2를 더하면 총합은 2, 다음에 4를 더해서 6, 다음에 6을 더해서 12, 다음에 8을 더해서 20, 그 다음 10을 더하면 30이 되는데 이때 30은 25보다 크므로 더한 뒤 바로 멈춘다. 따라서, 최종적으로 total에 저장된 값은 30이고, 함수는 30을 반환한다. 출력되는 값은 30이다.

문제 194. 정답: 4 **| 해설:** main()에서 print(unknown(3, 7))을 실행했다. 이때, unknown 함수는 두 매개변수 a와 b를 가지고 있다, a가 b보다 크면 a-b를, 아니면 b-a를 한 값을 리턴하는 구조이다. 현재 3은 7보다 작으므로 7-3의 계산값 4가 리턴된다. 따라서, 4가 출력된다.

문제 195. 정답: ① **| 해설:** 먼저, value = 1은 전역 변수이다. first() 함수 안에서 value = 10이라고 새로 만들었지만, 이것은 함수 안에서만 쓰는 지역 변수이고 전역 변수와는 다르다.
second() 함수에서는 global value라고 했기 때문에 전역 변수 value를 사용하게 된다. 처음 전역 변수는 1이었고 거기에 5를 더해서 6이 된다. 그래서 second: 6이 출력된다.
다시 first 함수로 돌아오면, 그 안의 지역 변수 value는 여전히 10이므로 first: 10이 출력된다. 마지막 줄에서 전역 변수 value를 출력하므로 global: 6이 된다. 따라서 실행 결과는 second: 6, first: 10,

global: 6 순서로 출력된다.

문제 196. 정답: ③ **| 해설:** [3] 함수 내부에서 전역 변수 값을 변경하려면 반드시 global 키워드를 사용해야 한다. (O) [1] 지역 변수는 함수 내부에서만 사용된다. (X) [2] 전역 변수는 함수 외부에 선언되어 프로그램 전체에서 사용 가능하다. (X) [4] 함수 내부에 같은 이름의 변수가 있으면 지역 변수가 우선 적용된다. (X) [5] 전역 변수는 프로그램 전체에서 유지되어 함수가 끝나도 사라지지 않는다. (X)

문제 197. 정답: ④ **| 해설:** check 함수는 n % 2 == 0, 즉 2로 나누어 떨어지면 짝수로 판단하여 "abc"를 반환한다. 따라서, 입력값은 짝수여야 프로그램을 실행했을 때 "abc"가 출력된다.

문제 198. 정답: ③ **| 해설:** [3] 함수, 변수, 클래스 등은 모듈이고, 이를 여러 개 모아 기능 단위로 구성한 것이 라이브러리이다. (O) [1] 라이브러리가 여러 모듈을 모아 놓은 것이다. (X) [2] import random as r은 삭제가 아니고 random을 r로 부르겠다는(별명 사용) 의미이다. (X) [4] 함수의 수에는 제한이 있지 않다. (X) [5] 따로 설치하지 않아도 기본적으로 설치되어 있는 내장 모듈이 존재한다. import는 이미 설치된 모듈이나 라이브러리를 코드에서 사용하겠다고 알려주는 것이다. (X)

문제 199. 정답: ⑤ **| 해설:** [5] import는 이미 설치된 모듈을 불러오는 역할을 한다. (O) [1] import는 설치 기능이 없고, 라이브러리 사용 선언일 뿐이다. (X) [2] pip install은 외부 라이브러리 설치용이며, 내장 모듈은 이미 설치되어 있어 설치할 필요가 없다. (X) [3] random은 파이썬 표준 라이브러리에 포함되어 있어 이미 설치되어 있고, pip install 없이 바로 import할 수 있다. (X) [4] pip install은 설치만 할 뿐, 코드에 import를 자동으로 추가해주지는 않는다. (X)

8차시 리스트와 다차원 구조 [개념편 30쪽 / 문제편 85쪽]

문제 200. 정답: ○ **| 해설:** 파이썬 리스트는 여러 자료형을 혼합하여 저장할 수 있다.

문제 201. 정답: X **| 해설:** 슬라이싱은 끝 인덱스 전까지 포함되므로 인덱스 1과 2의 항목만 가져온다.

문제 202. 정답: ○ **| 해설:** append() 메서드는 리스트 마지막에 항목 하나만 추가한다.

문제 203. 정답: X **| 해설:** 2번 인덱스에 "A"가 추가되고, 그 뒤 항목들은 밀린다.

문제 204. 정답: ○ **| 해설:** 슬라이싱으로 동일한 시작과 끝 인덱스를 지정하면 해당 위치에 여러 항목을 삽입할 수 있다.

문제 205. 정답: X **| 해설:** 기본 정렬은 오름차순이며, 내림차순은 `.sort(reverse=True)`로 해야 한다.

문제 206. 정답: X **| 해설:** 리스트는 앞쪽부터 순서대로 비교하고, 뒤쪽이 다르더라도 앞에서 참인 것이 있으면 무조건 True이다.

문제 207. 정답: ○ **| 해설:** list5[1][3]은 list5 리스트의 두 번째 행의 네 번째 항목이다. 2차원 리스트는 [행][열] 형태로 인덱싱하며, 각 인덱스는 왼쪽에서부터 0부터 시작한다.

문제 208. 정답: `b.append(i*2)` **| 해설:** for i in a:에서 반복될 때마다 i 변수에 a 리스트의 원소가 하나씩 들어가므로 이를 이용해 반복문이 실행될 때마다 b 리스트 안에 곱하기 2한 값을 추가하면 된다

문제 209. 정답: ③ **| 해설:** `for num in numbers`에서 num 변수에 numbers 리스트의 각 항목이 반복이 실행될 때마다 저장되게 된다. 이때, 반복문에서 실행되는 코드는 초기값이 0인 total 변수에 값을 더하는 것이므로, 리스트의 모든 원소의 합을 구하는 프로그램임을 알 수 있다. 3 + 7 + 1 + 9 + 5 = 25 이므로 출력되는 결과는 25이다.

문제 210. 정답: ④ **| 해설:** [4] 리스트는 .sort() 메서드 또는 sorted() 함수로 정렬할 수 있으며, 오름차순과 내림차순 모두 선택할 수 있다.(정렬할 때는 기본적으로 오름차순이다) (X) [1]&[2] 리스트는 여러 개의 서로 같거나 다른 자료형의 값을 저장할 수 있는 자료형이다. (O) [3] 리스트의 요소는 인덱스를 통해 접근할 수 있다. (O) [5] 여러 개의 데이터를 변수에 넣고자 할 때, 각 데이터를 콤마로 구분하고, 대괄호로 감싼다. (O)

문제 211. 정답: ⓐ len(subjects) ⓑ subjects[0] **| 해설:** len() 함수로 리스트의 길이를 구할 수 있고, 리스트의 특정 인덱스 값은 인덱싱을 사용하여 구할 수 있다.

문제 212. 정답: ⑤ **| 해설:** 인덱싱의 경우, 존재하지 않는 위치인 경우 오류가 발생하는 반면, 슬라이싱은 끝 인덱스가 리스트 길이를 넘어가도 오류가 발생하지 않는다.

문제 213. 정답: ② **| 해설:** data[2] = 1 + 9 = 10가 되어 2번 인덱스, 즉 세 번째 값이 바뀐다.

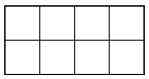
문제 214. 정답: -15 **| 해설:** 초기값이 10인 tot 변수에 리스트에 있는 항목을 각각 빼는 프로그램이므로 10-3-6-2-9-5=-15이다. 따라서, 출력되는 값은 -15이다.

문제 215. 정답: ④ **| 해설:** [4] 홀수 값들만 더하므로 7+1+9+3=20이 출력된다. (X) [1] data는 [4, 7, 1, 9, 3]로 5개의 요소가 들어있다. (O) [2] 홀수는 2로 나눈 나머지가 1이다. (O) [3] if 조건문 아래에 있으므로, 홀수일 때만 total에 더해진다. (O) [5] for n in data:는 리스트 data의 값을 순서대로 꺼낸다. (O)

문제 216. 정답: 25 **| 해설:** A[0][1] = 10, A[1][0] = 15이므로, 10 + 15 = 25이다.

문제 217. 정답: ③ **| 해설:** A[2]는 존재하지 않는다. 바깥쪽에 있는 리스트에는 세 번째 항목은 없다. 리스트는 0부터 수를 세므로 만약 두 번째에 있는 리스트를 의미하고 싶었다면 A[1]을 해야 한다.

문제 218. 정답: ② **| 해설:** for row in scores은 바깥쪽 리스트로부터 [90, 85]와 같은 안쪽 리스트를 가져온다. 이후, for score in row에서는 90과 같은 안쪽 리스트의 값을 가져온다. 이를 바탕으로 코드를 보면, 전체 리스트의 모든 숫자가 누적되어 최종 합이 계산되는 프로그램이다. 이때, 90+85+80+75+70+95 = 495 이므로 출력되는 결과는 495이다.

문제 219. 정답:  **| 해설:** 행은 가로줄, 열은 세로줄이다.

문제 220. 정답: ② **| 해설:** [2] append 메서드는 리스트의 끝에 새로운 항목을 하나 추가한다. (O) [1] insert 메서드는 특정 인덱스에 새로운 항목을 하나 추가한다.

문제 221. 정답: ⓐ append ⓑ insert **| 해설:** 기존 리스트 항목들 뒤에 'd'가 있으므로 append 메서드로 추가된 것이고, 인덱스 1번 자리에 'e'가 있는 것으로 보아 insert 메서드로 중간에 삽입된 것이다.

문제 222. 정답: [10, 30] **| 해설:** remove 메서드를 통해 값이 20인 항목이 삭제되고, pop 메서드를 통해 마지막 위치의 항목이 삭제되었다. pop 메서드의 경우, 인덱스를 제시하지 않으면 제일 마지막 인덱스로 판단된다.

문제 223. 정답: ③ **| 해설:** remove 메서드는 값을 통한 삭제, pop 메서드는 인덱스를 통한 삭제이다.

문제 224. 정답: ④ **| 해설:** .sort() 메서드와 달리, sorted() 함수는 원본 리스트는 건드리지 않고, 정렬된 새 리스트를 반환(리턴)한다.

문제 225. 정답: ③ **| 해설:** scores 리스트에는 [80, 60, 90]가 있다. new_scores에는 sorted 함수와 그 속에 reverse=True가 있으므로 scores의 복사본을 내림차순으로 정렬한 새 리스트가 저장된다. 그리고, scores.sort()는 scores 리스트를 기본값인 오름차순으로 정렬한다.

문제 226. 정답: [100, 10, 15, 200] **| 해설:** [5, 10, 15]에서 인덱스 1 자리에 100이 삽입되어 [5, 100, 10, 15]가 되고, 끝에 200이 추가되어 [5, 100, 10, 15, 200]가 되고, 5가 삭제되어 최종적으로 [100, 10, 15, 200]이 되었다.

문제 227. 정답: [3, 1, 4, [1, 4]] **| 해설:** 리스트를 append 한다고 해서 여러 값이 리스트에 여러 항목으로 나누어 추가되는 것이 아니라, 리스트 자체가 항목으로 추가된다.

문제 228. 정답:

```
nums = [1, 5, 3, 7, 2, 4]
result = []
for n in nums:
    if n > 3:
        result.append(n)
result.sort()
print(result)
```

| 해설: for문에서 리스트를 사용하면 리스트의 각 값이 for 반복문의 1회 실행 때 마다 하나씩 들어가게 된다. append

메서드는 result 리스트 오른쪽 끝에 항목을 추가해주고, sort 메서드는 리스트를 오름차순(1,2,3,...) 정렬해준다.

문제 229. 정답: [[10, 15, 20], [30]] **| 해설:** `matrix[0]`은 [][]에서 처음에 있는 대괄호이므로 바깥 리스트의 첫 번째 값 `[10, 20]`를 의미한다. `[10, 20]`의 1번 인덱스 위치에 15를 삽입하면 `[10, 15, 20]`이 된다. `matrix[1]`은 바깥 리스트의 두 번째 값 `[30, 40]`을 의미한다. 여기서 40을 삭제하면 `[30]`이 된다.

문제 230. 정답: ③ **| 해설:** .pop()은 리스트의 마지막 요소를 제거한다. 리스트에 바로 적용이 된다.

문제 231. 정답: [10, 15, 30] **| 해설:** 초기에 nums 리스트는 [10, 20, 30]이다. 여기서 append(40)를 하여 [10, 20, 30, 40]이 되고, insert(1, 15)를 하여 [10, 15, 20, 30, 40]이, pop()으로 [10, 15, 20, 30]이, remove(20)으로 [10, 15, 30]이 된다.

문제 232. 정답: ② **| 해설:** ㄱ 제시된 프로그램은 각 리스트별 최고 점수를 result에 저장하는 구조이다. (O) ㄹ result.append(max_score)는 구한 최고 점수(max_score 변수)를 리스트에 추가(append 메서드)하는 명령이다. (O) ㄴ scores[i][j]는 i번째 항목 안의 j번째 항목을 의미한다. (X) ㄷ 각 열에서의 최고 점수를 저장한다. (X)

문제 233. 정답: ['grape', 'banana', 'melon'] **| 해설:** 초기에 fruits 리스트는 ["apple", "banana", "cherry"]이다. 여기서 insert(1, "grape")를 하여 ["apple", "grape", "banana", "cherry"]가 되고, remove("apple")을 하여 ["grape", "banana", "cherry"]가 되며, append("melon")을 하여 ["grape", "banana", "cherry", "melon"]이 되고, 마지막으로 pop(2)를 실행하여 "cherry"가 제거되며 최종적으로 ["grape", "banana", "melon"]이 된다.

9차시 문자열과 시퀀스 [개념편 37쪽 / 문제편 99쪽]

문제 234. 정답: ○ **| 해설:** 문자열은 시퀀스 자료형에 속하기 때문에 리스트처럼 인덱싱이 가능하다.

문제 235. 정답: X **| 해설:** 슬라이싱은 끝 범위를 포함하지 않으므로, 실제로는 인덱스 2~4만 출력된다.

문제 236. 정답: X **| 해설:** 공백이 하나라도 포함되면 False를 반환한다. 숫자만 있어야 True다.

문제 237. 정답: ○ **| 해설:** count는 문자열 내에서 특정 문자가 몇 번 나오는지 정수값으로 알려주는 메서드이다.

문제 238. 정답: X **| 해설:** .strip()은 문자열의 양쪽 끝 공백만 제거하므로 내부 공백은 그대로 남는다.

문제 239. 정답: ○ **| 해설:** len()은 시퀀스의 길이를 구하는 함수로, 문자열과 리스트 모두 시퀀스에 속하는 자료형이므로 적용할 수 있다.

문제 240. 정답: ⑤ **| 해설:** .upper()는 어떤 대소문자 입력이든 모두 대문자로 바꿔준다.

문제 241. 정답: ① **| 해설:** isalpha는 숫자 때문에 False, isdigit도 문자 포함되어 False, islower는 대문자가 포함되어 있어 False이다.

문제 242. 정답: ② **| 해설:** 문자열은 값이 바뀌면 완전히 새로운 문자열로 바뀌는 것이지, 기존의 문자열 시퀀스를 수정하는 것이 아니기 때문에, strip 같은 메서드로 처리했을 때 기존 내용이 변경되는 것이 아니라, 변경된 문자열이 리턴된다. 그래서 마지막에는 여전히 #가 남아있던 것이다.

문제 243. 정답: strip **| 해설:** strip 메서드는 문자열의 양 끝에 있는 특정 문자를 삭제해준다.

문제 244. 정답: one,two,three **| 해설:** "one/two/three"를 / 기준으로 잘라 리스트로 만든 후, 쉼표(,)를 사이에 넣고 여러 항목을 하나의 문자열로 합쳤다.

문제 245. 정답: replace("JAVA", "Python", 1)은 첫 번째 "JAVA"만 바꾸고, 나머지 "JAVA"는 남겨둔다. 그 후 다시 replace("JAVA", "Python")을 적용해서 나머지 하나도 바꾼다. 결과적으로 두 번 바뀌어 "I like Python and Python"이 된다. **| 해설:** replace에서 개수를 비워놓으면 문자열 전체의 단어를 바꾸고 숫자를 적으면 그 개수만큼만 바꾸기 한다.

문제 246. 정답: ② **| 해설:** strip()은 양쪽 공백 제거이다. "apple"은 문자열 안에 2번 등장하기 때문에 count("apple")은 2를 반환한다.

문제 247. 정답: cd **| 해설:** "a b c d e f g"는 리스트로 나타내면
`['a', ' ', 'b', ' ', 'c', ' ', 'd', ' ', 'e', ' ', 'f', ' ', 'g']` 이다.

11차시

객체와 클래스 [개념편 46쪽 / 문제편 104쪽]

문제 248. 정답: ○ **| 해설:** 객체는 프로그램 속에서 현실 세계의 대상을 코드로 표현한 것으로, 속성과 기능을 가질 수 있다.

문제 249. 정답: X **| 해설:** 객체는 속성이나 기능 중 하나만 가질 수도 있다.

문제 250. 정답: ○ **| 해설:** __init__은 객체가 생성될 때 자동으로 실행되며, 속성을 초기화하는 데 사용된다.

문제 251. 정답: ○ **| 해설:** 인스턴스는 메서드를 호출할 때 자동으로 자기 자신을 인수로 넘기므로 self가 첫 번째 매개변수로 있어야 한다.

문제 252. 정답: X **| 해설:** 클래스 하나로 여러 개의 인스턴스를 만들 수 있다.

문제 253. 정답: ○ **| 해설:** 클래스를 정의할 때는 반드시 class 클래스이름: 형식을 사용한다.

문제 254. 정답: ○ **| 해설:** 속성(변수)과 메서드(함수)를 함께 정의할 수 있는 것이 클래스의 핵심 기능이다.

문제 255. 정답: ④ **| 해설:** 객체는 현실 세계의 사물이나 개념을 코드로 표현한 실체이며, 속성과 기능을 포함할 수 있다. 클래스는 객체를 만들기 위한 틀이며, 객체는 반드시 클래스 기반으로 생성된다.

문제 256. 정답: ③ **| 해설:** 인스턴스는 클래스라는 설계도를 기반으로 생성된 객체이며, 속성은 개별 인스턴스에, 기능은 클래스에 존재한다.

문제 257. 정답: ⑤ **| 해설:** 학생 C 인스턴스는 클래스의 '다른 이름'이 아니라, 클래스에서 만들어진 '객체 하나'를 의미한다.

문제 258. 정답: ④ **| 해설:** self는 클래스가 아니라, 생성된 인스턴스 자기 자신을 의미한다.

문제 259. 정답: ⑤ **| 해설:** 생성자 내부에서도 새로운 속성을 정의할 수 있다. 통상적으로 속성 초기화를 담당할 뿐, 새로운 속성을 만들 수 있다.

문제 260. 정답: ③ **| 해설:** <프로그램>에서 show() 메서드는 인스턴스의 name과 score 속성을 출력한다. 생성 시 "정윤"과 90을 인자로 전달했기 때문에 값이 그대로 출력된다.

문제 261. 정답: 동수 **| 해설:** self.name에 값이 저장되고, print(s.name)을 통해 그대로 출력된다.

문제 262. 정답: 20 **| 해설:** 초기값 10에서 double 메서드 호출 후 곱하기 2 되어 20이 되었다.

문제 263. 정답: 8 **| 해설:** self.a가 change() 호출로 인해 5에서 8로 바뀌었다.

QR 코드를 찍어서
자동채점기를 이용해 보세요!