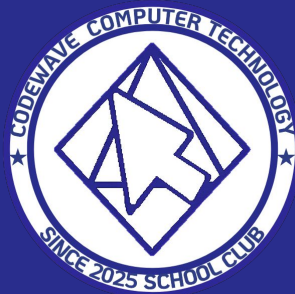
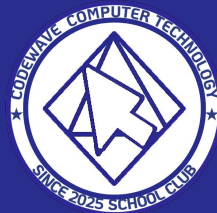


# 코드웨이브 OT

지금까지 이런 동아리는 없었다



## 코드웨이브의 역사



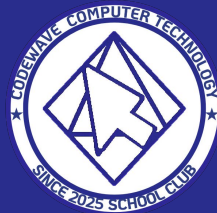
코드웨이브는 2025년에 신설된 동아리로 현재까지도 역사를 써내려가고 있습니다.

2025년 3월 19일 - 1기 멤버 선별

2025년 6월 13일 - 1대 동아리 부장, 차장 선발 (류용헌, 신지욱)

2026년 2월 - 1기 3학년 멤버 졸업

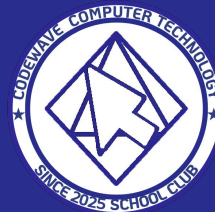
2026년 3월 - 2기 멤버 선별



코드웨이브에는 꽤나 장점이 많습니다.

첫번째 장점으로서는 자유로움이 첫번째 장점이고,  
두번째 장점으로서는 학생들이 참여하는 분위기가 강하다는게 장점입니다.  
세번째 장점은 현 부장이랑 차장이 IT에 진심이라서 많은 전문 지식을 배워갈 수 있다는 점입니다.

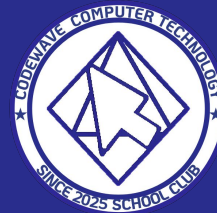
솔직히 이정도만 해도 다른 동아리 뛰어넘습니다~ 하하



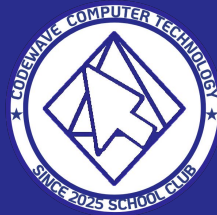
코드웨이브의 커리큘럼은 이러한 방식으로 짜여있습니다.

구분	전공 과목					
전공 필수	컴퓨터 작동과 정	프로그래 밍 개념	파이썬 프로그래 밍	객체지향 프로그래 밍 개념	정보보 안 기초	자료구 조
전공 심화	C 프로그래밍 기초			백준 C언어 기초 문제 해석		

Q&A



저희 이제 뭐해요?



저희는 넓은 IT지식을 가지기 위해 지금부터 **프로그래밍**이라는 것을 할거예요~



지금 학생들 표정

**프로그래밍** 즉 **코딩**은 컴퓨터에 원하는 작업을 지시하고 부여하기 위해 필요한 명령을 작성하고, 그 명령어들을 **모은 것**이라고 할 수 있습니다.

이렇게 만들어진 프로그램들이 모여 소프트웨어가 되기 때문에, 프로그래밍은 소프트웨어를 생성하는 과정이라고도 할 수 있습니다.



코딩은 무엇으로 하죠?

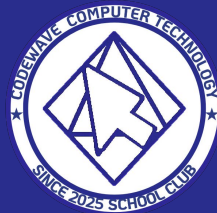
우리가 외국어를 익히지 않으면 외국인이랑 대화를 못하는 것처럼  
**컴퓨터 언어**를 못 익히면 컴퓨터랑은 대화를 할 수가 없잖아요?

근데 우리가 컴퓨터 언어를 언제 외워서 언제 써먹어요~

그래서 나온게 **프로그래밍 언어**(컴퓨터 언어 번역기)입니다.







## 프로그래밍 언어의 역할

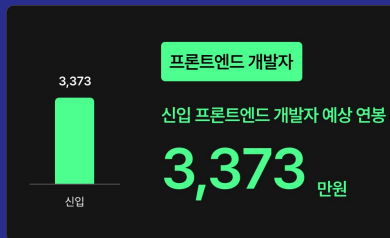
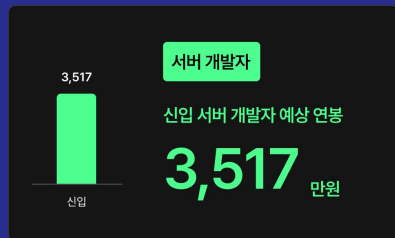
프로그래밍 언어에도 각각의 암묵적인 역할이 있습니다.

대표적으로 나뉘는것이 ‘**프론트엔드**’와 ‘**백엔드**’라는 것인데

쉽게 설명하자면

프론트엔드는 유저의 눈에 보이는 부분을 뜻하고 (사이트)  
백엔드는 유저의 눈에는 안보이지만, 내부 서버를 처리하는  
아주 중요한 부분을 뜻합니다. (사이트 내부 데이터 처리)

프론트엔드랑 백엔드를 동시에 개발하는 개발자를 우리는 풀스택 개발자라고 부르는데  
돈을 엄청나게 벌겠죠? 아래 단일 개발자 연봉 사진만 봐도 높아보이는데



## 인터프리터 VS 컴파일러

### 컴파일러 언어란?

프로그램의 모든 명령어를 0과 1로 변환해서 실행하는 언어입니다.  
(즉 컴퓨터가 알아듣기 쉽다는 소리입니다.)



### 인터프리터 언어란?

한 줄씩 읽어서 바로 실행하는 언어입니다.  
(우리가 영어를 한 줄씩 해석해서 읽는 것과 똑같습니다.)

이 2개를 봤을때 솔직히 컴파일러가 더 빠르겠죠?  
네 맞습니다.



## 프레임워크

프레임워크는 개발을 할때 기본적인 뼈대와 규칙을 미리 제공해 주는 그냥 아주 개발에 도움이 되는 그런 틀입니다.

대표적인 프레임워크로는



Unity



UNREAL  
ENGINE



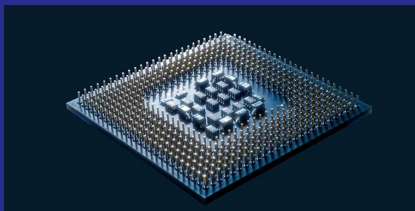
자바 , 유니티 , React JS , Node . js 등이 있고  
C# , C++ , Javascript 등의 언어를 사용합니다.

그래서 프로그래밍으로 뭘 해요?

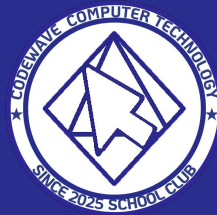
결론적으로 **프로그래밍**으로 무엇을 할수 있냐면?

반도체, 게임, AI 등 소프트웨어에 관련된 모든 것들을 만들어 볼 수도 있고  
정보보안 , 컨설팅 쪽의 분야로도 손을 뻗어볼 수 있습니다.

위 라인업을 볼때 **21세기 최고의 직업**은 프로그래머라고 손꼽을 수 있겠죠?  
라고 단정 짓지는 않겠습니다~



프로그래밍 그거 어떻게 하는건데요?



지금부터 저희는 **파이썬**이라는 언어를 쓸거예요~



C언어 보다는 완전 쉬운 언어니깐 저만 따라오면 고수가 될 수 있다~

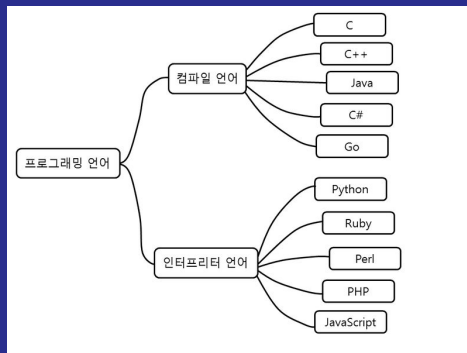
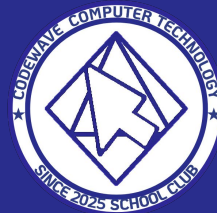
## 파이썬 이해하기

```
1 print("Hello world!")
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("Hello world!");  
4     }  
5 }
```

위 그림처럼 파이썬이란 배우기 쉽고 직관적인 문법을 가진 프로그래밍 언어입니다. 덕분에 초보자도 빠르게 학습할 수 있습니다. 웹 개발, 데이터 과학, 인공지능, 자동화 등 다양한 분야에서 활용되며, 이를 돕는 방대한 라이브러리와 프레임워크를 자랑합니다~

## 인터프리터 언어



**파이썬은 인터프리터 언어입니다.**

**인터프리터 언어란 코드를 한 줄씩 읽고 바로 실행하는 언어로, 실시간 실행을 통해 컴파일러 언어에 비해 빠르게 테스트가 가능하고 별도의 번역(컴파일) 과정 없이 실행할 수 있습니다.**

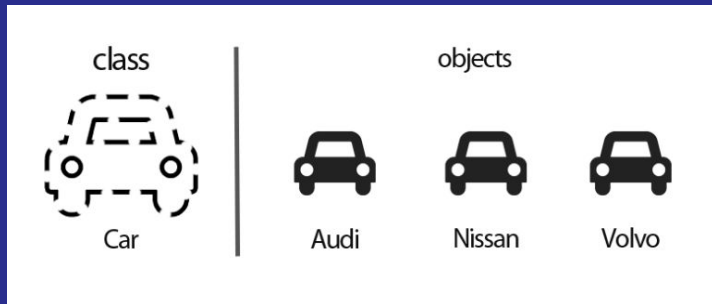
**다만 한 줄씩 실시간으로 번역하기 때문에 컴파일러 언어에 비해 느리다는 단점이 있습니다.**

## 객체 지향 언어

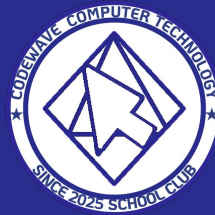
파이썬은 심지어 객체 지향 언어로도 알려져 있습니다.  
객체 지향 언어란 객체(Object)를 중심으로 프로그램을 구성하는 방식입니다.

자동차로 비유 하자면 아래 그림과 같이 클래스는 자동차의 설계도라고 정의 할 수 있는데, 이 클래스에는 자동차의 기본적인 특징(바퀴 수 등)이 포함되어 있습니다.

그리고 그 클래스를 이용하여 만들어진 차들(인스턴스)들은 클래스에서 정의한 기능들(메서드)를 사용할 수도 있으면서도 고유한 데이터 값을 가지고 있는 것이 객체지향 프로그래밍의 핵심입니다.







# - 기초적인 주석 처리 방법이다.

print() - 표준 입출력 함수에서 출력을 담당한다.

input() - 표준 입출력 함수에서 입력을 담당한다.

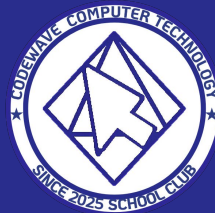
int() - 특정한 데이터를 정수형 형태로 만들어 주는 함수이다.

float() - 특정한 데이터를 실수형 형태로 만들어 주는 함수이다.

str() - 특정한 데이터를 문자열 형태로 만들어 주는 함수이다.

bool() - 이하 생략

len() - 길이를 구하는 함수(문자열, 리스트 등)



자료형은 프로그래밍에서 데이터의 종류와 그 데이터를 어떻게 저장하고 처리할지를 정의하는 개념입니다.

기본 자료형에는 int , float , str , bool 형이

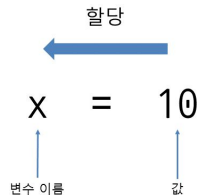
컬렉션 자료형에는 list , tuple , dict , set 형이

특수 자료형에는 NoneType , bytes , bytearray 형이

고급 자료형에는 range , frozenset , complax 형이 있습니다.

변수는 데이터를 저장하는 이름입니다.

값을 메모리에 저장하고 나중에 그 이름을 불러서 값을 사용할 수 있습니다.  
파이썬은 변수의 자료형을 미리 명시하지 않아도 되기 때문에 편리합니다.



The diagram shows the assignment statement `x = 10`. A blue arrow labeled '할당' (Assignment) points from the value '10' to the variable 'x'. Below 'x', a blue arrow labeled '변수 이름' (Variable name) points to the variable. Below '10', a blue arrow labeled '값' (Value) points to the value.

파이썬에서 변수를 선언할 때는 규칙이 있습니다.

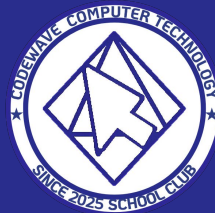
첫째, 변수 이름은 영문,숫자,\_로만 구성할 수 있습니다.

둘째, 숫자로 시작하면 안됩니다.

셋째, 같은 변수에 다른 값을 넣으면 자료형이 바뀔 수 있습니다.

```
1 MyName = 1
```

## 연산자



연산자는 연산을 수행하기 위해 사용되는 특수 기호 또는 키워드입니다.  
연산자는 주로 수치 계산, 비교, 논리 판단등의 다양한 작업을 수행하는 데 사용됩니다.

대입 연산자 : 변수에 값을 대입합니다.

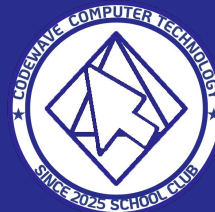
산술 연산자 : 사칙연산과 같은 기본적인 수학적 연산을 수행합니다.

비교 연산자 : 두 값의 대소 관계에 따라 참(true) 또는 거짓(false)를 판단합니다.

논리 연산자 : 하나 또는 두 개 이상의 조건을 비교하여 참 또는 거짓을 판단한다.

연산자	종류
대입 연산자	=, +=, -= 등
산술 연산자	+, -, *, /, //, %, **
비교 연산자	==, !=, >, <, >=, <=
논리 연산자	and, or, not

## 인덱싱



인덱싱은 리스트, 튜플 등 순서가 있는 자료형에서 특정 위치에 있는 단일 요소를 추출하는 과정입니다.

이때 컴퓨터는 1이 아니라 0부터 세기 때문에 인덱싱은 0부터 시작합니다.

인덱싱을 사용하는 방법은 [] 대괄호 안에 추출할 위치에 있는 값의 인덱싱 번호를 대입해주면 됩니다.

j	i		m	o	o		i	s		K	o	r	e	a	n				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

## 조건문

```
if (조건):  
    수행할 문장  
else:  
    수행할 문장
```

파이썬 조건문은 프로그램에서 특정 조건에 따라 서로 다른 코드 블록을 실행하도록 제어하는 구문입니다.

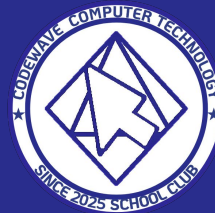
조건문이 참(true)인지 거짓(false)인지에 따라 코드의 흐름을 분기시키며 주로, if , elif , else 키워드를 사용합니다.

파이썬에서는 조건문을 사용할 때 세 가지로 나누어 조건을 판단합니다.

if : 조건식의 참/거짓을 판단하여 만일 참이라면 코드를 실행하고 조건문을 벗어납니다.  
elif : if의 조건이 참이 아니고, elif의 조건이 참이라면 코드를 실행하고 조건문을 벗어납니다.  
else : 모든 조건이 참이 아닐때만 코드를 실행하고, 조건문을 벗어납니다.

다만, elif나 else를 사용할 때에는 위에 if문이 있어야만 작동합니다.

## 반복문



반복문은 말 그대로 특정 코드를 여러 번 실행하도록 하는 구문입니다.

마치 우리가 어떤 일을 반복해서 해야 할 때 일정한 패턴을 반복하는 것처럼 프로그램에서도 같은 코드를 여러 번 실행해야 할 때 반복문을 사용합니다.

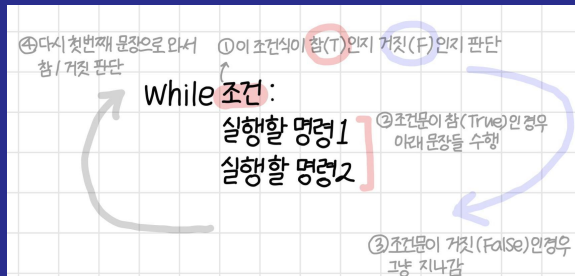
파이썬 반복문은 특정 코드 블록을 여러 번 실행하도록 제어하는 구문으로, 프로그램에서 반복적인 작업을 처리할 때 사용됩니다. 반복문은 데이터를 순차적으로 처리하거나 조건이 만족될 때까지 작업을 반복하는데 유용합니다.

파이썬에는 두 가지 종류의 반복문이 존재하고 있습니다.

**while문** : 조건을 기준으로 반복합니다.  
**for문** : 반복 범위를 기준으로 반복합니다.



## 반복문 (while문)



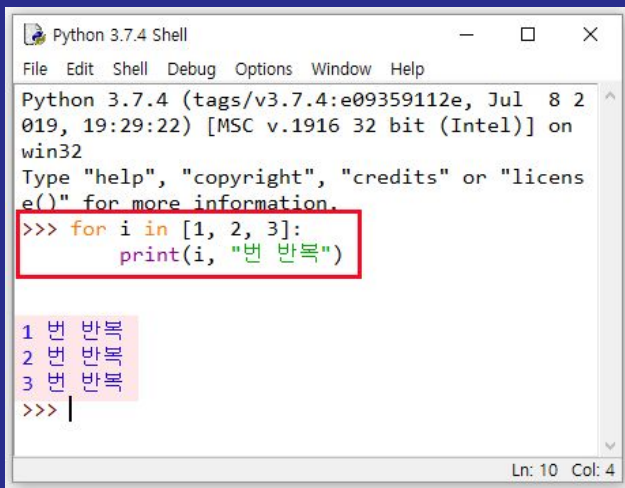
while문은 주어진 조건식이 참(true)인 동안 계속해서 반복 작업을 수행합니다.

조건이 거짓(false)이 될 때까지 반복되므로, 반복 횟수가 명확하지 않거나 특정 조건을 만족할 때까지 반복해야 할 때 유용합니다. while문은 반복 횟수가 가변적이거나 조건에 따라 반복 여부가 결정될 때 사용됩니다.

## 반복문 (for문)

```
for(반복 변수) in(반복 범위):
    반복할 코드
```

for문 반복 범위에 있는 값이 반복 변수에 저장되어 반복하여 코드가 실행됩니다.  
반복 범위에는 컬렉션과 range 자료형을 사용할 수 있습니다.



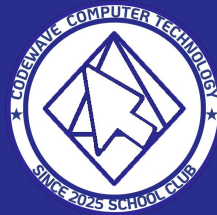
```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> for i in [1, 2, 3]:
    print(i, "번 반복")
1 번 반복
2 번 반복
3 번 반복
>>> |
```

```
for i in range(1,10):
    print("8 x %d=%d"%(i, 8*i))

...

dan=int(input("단 입력:"))
for i in range(1,10):
    print("%d x %d"%(dan,i,dan*i))
```

수고하셨습니다



축하합니다~ 오늘 당신은 **프로그래머**에 2% 가까워졌습니다!

앞으로도 **전진**하시죠