

# 1 세상 변화를 즐기기 위한 도구

[Open in Colab](#)

"우리 모두가 컴퓨터 전문가가 될 필요는 없지만, 디지털 기기 없이 살수 없는 시대이기 때문에 컴퓨팅 사고력과 디지털 역량이 생존경쟁인 시대"

"세상과 분리되서 살지 않는 한.. 우리 스스로는 디지털 기반 데이터를 끊임없이 만들어 무료로 제공하고 있기 때문에 데이터로 움직이는 세상의 흐름을 모른다면 20세기 세대들과 다를게 없거나 금방 끝내(?)가 될 수밖에 없는 시대"

"데이터를 만드는건 우리 스스로이기 때문에, 정보 노출의 위험을 다른 사람 책임이라 할 수 없고 우리 스스로의 책임이라 할 수 있으며 그 책임을 이해하는 것이 데이터사이언스"

## 1.1 현업에서의 데이터사이언티스트

1) 데이터분석 업무를 하는 데이터사이언티스트는 혼자가 아니라 팀으로 일한다!

- 학교에서나 혼자하지 학교 밖은 혼자 할수 있는 것이 전혀 없음
- 갖춰야 하는 스킬이 너무나 방대하기 때문에 팀을 구성하여 업무를 진행
  - 수많은 돈을 보유한 기업의 오너나 사장도 혼자 일하지 않음
  - 정말 혼자 일하는 사람은.. 망해가는 곳에 있거나 백수일 때



(Data Science Team)

2) 데이터사이언티스트 팀 구성원들은 각자의 전문성에 집중하며 협업!



### Data Analyst (DA)

: Assist DS with domain understanding, data preprocessing and problem defining.



### Data Scientist (DS)

: Prepares data, engineers features, most valuable skill: training models.



### Data Engineer (DE)

: Data acquisition focus. Build data pipelines. Not uncommon to have 5:1 ratio DE:DS

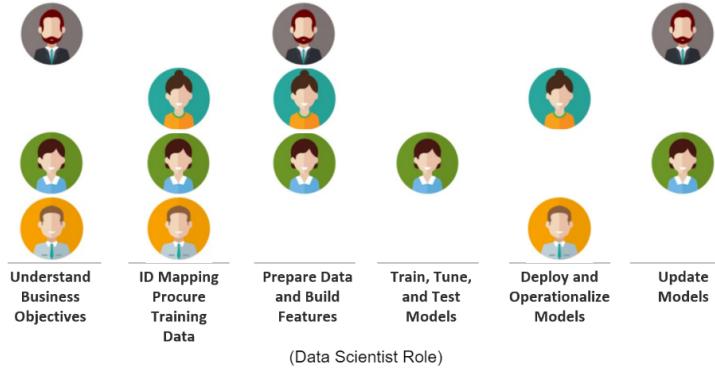


### Data Application Architect (DAA)

: Design complete solution; deploy and maintain models in production

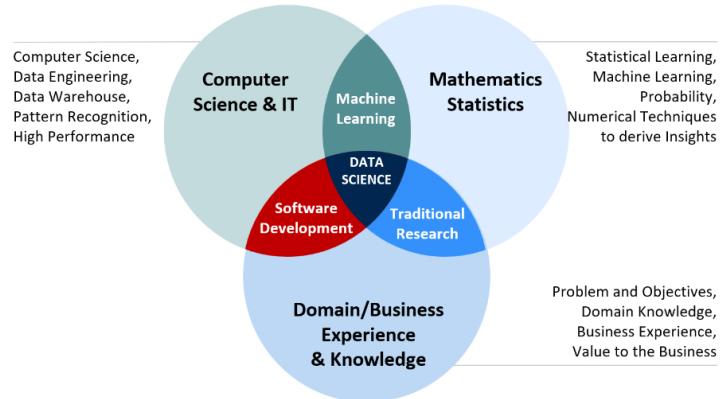
#### ➤ Typical Collaboration of Data Science Project

- Makes data science teams more productive
- Broad support for open source libraries in various languages



3) 데이터사이언스의 역할은 과거에도 있었지만 점점 진화하고 있음!

"세상이 변화하듯 직업의 영역도 변화하고 겹치는 부분도 많아짐"



- 데이터 사이언티스트 = 데이터 분석가
- 컴퓨터 디벨로퍼 = 프로그래머 = 개발자 = 엔지니어 = 데이터 엔지니어
- 데이터 애널리스트 = 애널리스트

	Data Analyst	Analyst	(정통) 기획 및 전략	- 기획서, 보고서, 기초분석, 시각화
		Data Analyst	데이터분석 설계	- 데이터 표준 확립, 구조 및 품질관리, 데이터 분석
	Data Scientist	Statistician	(정통) 통계기반 연구	- 결과가 나온 이유를 연구하는 방향
		Data Scientist	데이터분석 전반 연구	- Why보다 Result 마인/태용/전략/의사결정 방향
	ML / DL Engineer	ML / DL Engineer	응용 및 구현 연구	- 연구결과를 실질적 서비스 및 비즈니스 접증화
		Database Administrator	데이터베이스 관리자	- 데이터베이스 운영, 관리, 설계
	Data Engineer	Back-end Engineer	(정통) 백엔드 개발자	- 서비스 개발, 데이터베이스 시스템 구현 및 관리 - RDB/NoSQL/Hbase/Spark 등
		Infra Engineer	인프라 엔지니어	- 데이터 파이프라인 구축 및 운영 - Cloud/Spark/Hadoop 등

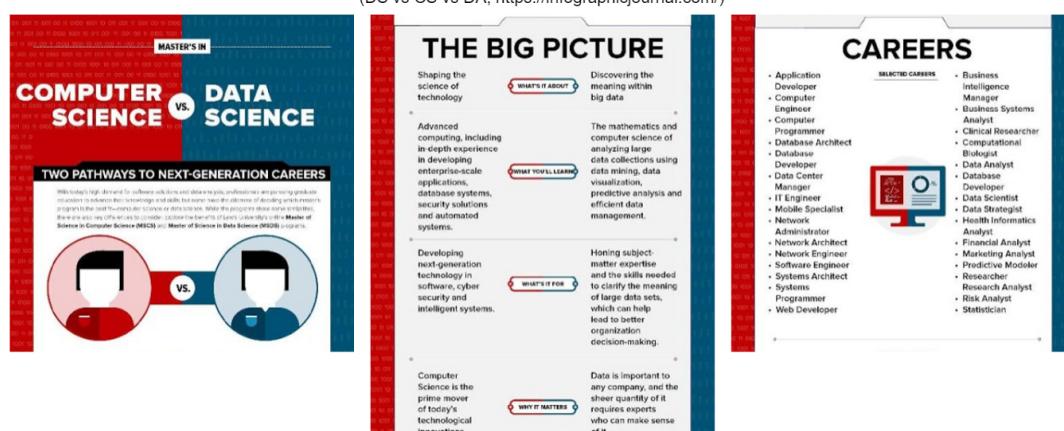
(Evolving Role of Data Scientist)

4) 데이터사이언티스트 vs 컴퓨터디벨로퍼 vs 데이터애널리스트

"세상이 변화하듯 직업의 영역도 변화하고 겹치는 부분도 많아짐"

- 좋게 얘기하면 변화지만, 현실적으로 상대의 영역을 흡수/문어발/뺏는 것
- 흡수하여 확장/진화하는 사람들은 살아남고 몸값이 높아지지만, 그렇지 않은 사람들은 자리를 뺏기고 몸값은 하락함



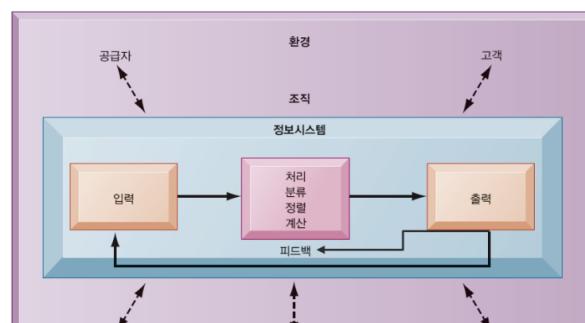


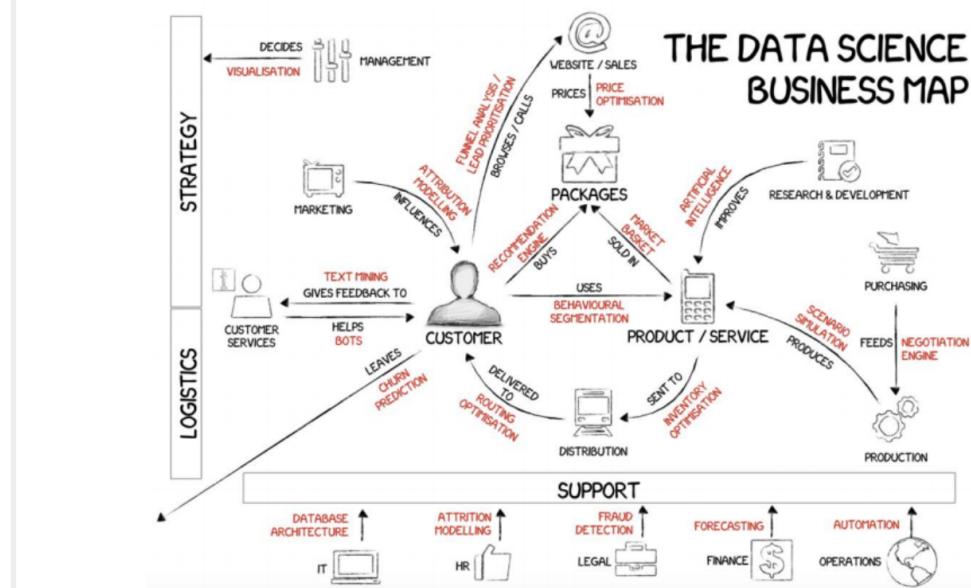
## 1.2 비즈니스/사람/컴퓨터의 작동방식

"비즈니스 작동요소 = 조직 작동방식 = 사람 교류방식 = 컴퓨터 교류방식"

1) 비즈니스 작동요소: 입력(x)/처리(f)/출력(y)/검증(e)

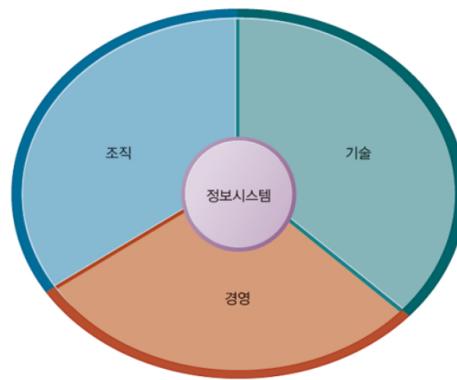
- **입력(Input):** 조직 내부 또는 외부 환경으로부터 원시(Raw) 데이터를 획득하거나 수집
- **처리(Processing / Function):** 원시 데이터를 의미 있는 형태로 변경
- **출력(Output):** 처리된 정보를 사용할 사람이나 활용할 활동에 전달
- **검증(Feedback / Evaluation):** 작동요소를 평가하거나 교정하기 위한 성능지표(담당자의 평가, 회사의 방향 등)





2) 조직 작동방식: 입력(x)/처리(f)/출력(y)/검증(e)

- 현명한 비즈니스 의사결정을 위해 기업은 경영진 + 조직 + **기술**을 끊임없이 변화



3) 사람 교류방식: 입력(x)/처리(f)/출력(y)/검증(e)

- 밥먹자(x)/어쩌지?(f)/싫은데(y)/왜?(e)
- 밥먹자(x)/어쩌지?(f)/좋아(y)/젠장(e)

4) 젠장(검증)



1) 밥먹자(입력)  
3) 좋아( 출력 )

2) 어쩌지(처리)

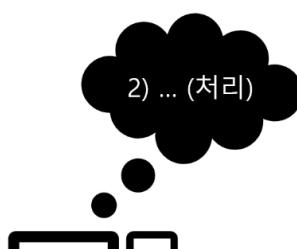


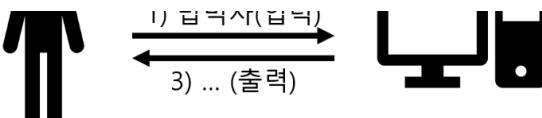
4) 컴퓨터 교류방식: 입력(x)/처리(f)/출력(y)/검증(e)

4) 좋아(검증)



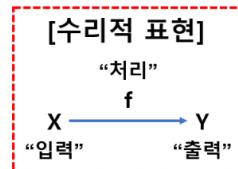
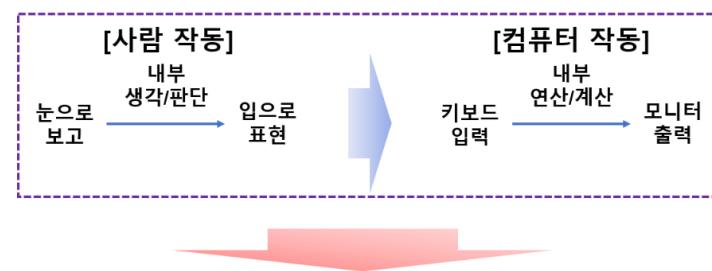
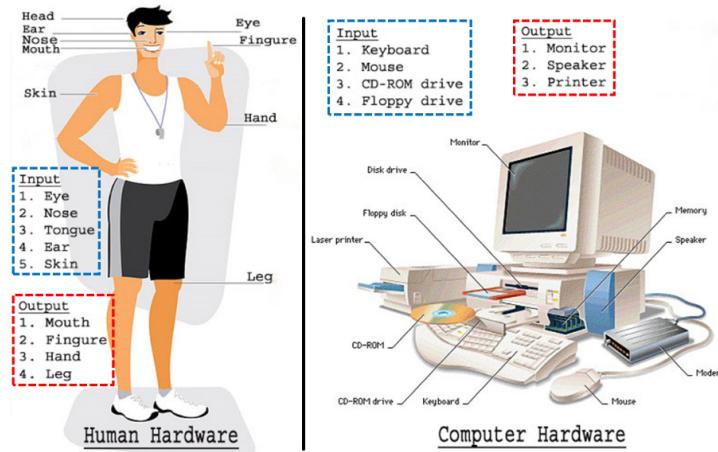
2) ... (처리)





### 1.3 컴퓨터의 구조와 프로그램

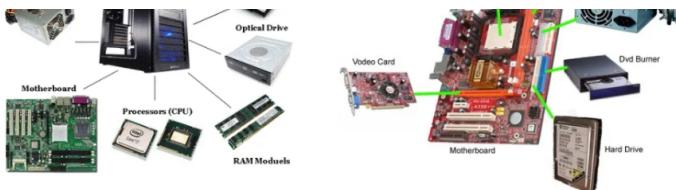
"사람과 컴퓨터의 교류방식이 유사한 이유는, 사람을 모방하여 컴퓨터를 설계했기 때문"



1) 하드웨어(Hardware): 데이터 처리/계산, 그래픽 표현, 응용작업 등 사용자 목적으로 맞는 지정된 작업을 수행하도록 설계된 물리적으로 존재하는 전자 또는 기기 제품

- **입력장치:** 사용자가 컴퓨터 시스템과 상호작용하고 사용자의 입력데이터를 받아들이는 기기
- **출력장치:** 사용자가 처리한 데이터를 표시하거나 저장하는 여러 유형의 기기



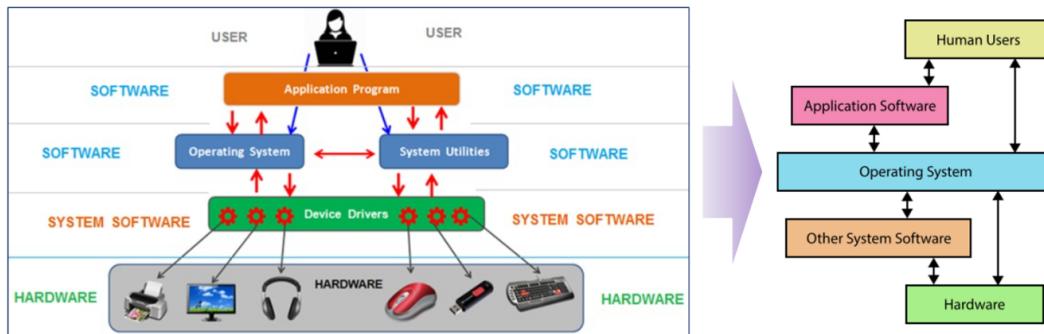


**Computer CPU and Motherboard Hardware Components**

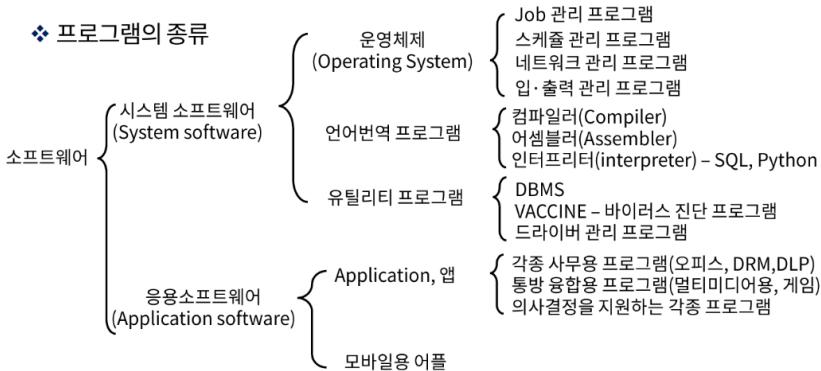
- 2) 작동구조(Architecture): 수많은 프로그램(소프트웨어)들이 결합되어 하드웨어를 운영

- 소프트웨어:

- 사람과 컴퓨터기반 하드웨어를 소통하게 하는 **프로그램**
- 하드웨어의 구성요소를 작동하고 시스템에서 사용자가 지정한 작업을 수행하는데 필요한 **프로그램**



- 시스템 소프트웨어: 작동하는 다양한 하드웨어 구성요소 및 시스템에 연결된 하드웨어 장치들을 제어하기 위한 **프로그램**
- 응용 소프트웨어: 사용자가 원하고 지정한 다양한 작업을 수행하기 위한 **프로그램**



- 앱(App)과 프로그램:

- App은 \*\* Application Program \*\*의 줄임말
- 모바일 앱은 소프트웨어/프로그램의 한 종류이자 다른 표현
- 앱은 곧 프로그램!

3) 결론:

- 사람이 원하는 작업을 컴퓨터에게 요청/말하기 위해서 다양한 **프로그램**을 사용
- 다양한 프로그램은 알아서 다양한 하드웨어들을 운영하여 사람이 요청/말한 것에 답변을 함

## 2 변화를 즐기기 위한 프로그래밍

### 2.1 프로그램과 프로그래밍 언어

- 1) 코딩(Coding)과 프로그래밍(Programming): 코딩으로 컴퓨터에게 말을 건다

코드(Code)	코딩(Code + ing)
컴퓨터에게 말을 걸기 위해 쓴 글	
프로그램(Program)	프로그래밍(Program + ing)
특정 작업/목적을 수행하는 코드(Code)의 모음	

- 2) 프로그램(Program = Software): 동일한 말을 반복하기 귀찮으니 반복되는 말은 프로그램으로 만든다

"특정 작업/목적을 수행하기 위해 만든 것으로 여러 **코드**를 모은 후, 사람이 이해할 수 있는 **UI (User Interface)**를 덧어 씌워서 생성"

- 예시:

프로그램 이름	주 생성목적
휴지통	필요없어서 버린 자료 보관
Word	문서 작성
한글	문서 작성
Excel	수치 계산, 문서 작성, 각종 업무
Photoshop	그래픽 이미지 생성 및 처리
Internet, Chrome..	인터넷 환경 접속 및 정보 처리
캡처 도구	화면을 이미지로 저장
TeamViewer	원격으로 기기제어
카카오톡	커뮤니케이션, 공유
스마트폰	커뮤니케이션, 수치계산, 문서작성, 운동지원, ...
자율주행차	자동으로 운전

### 3) 프로그래머(Programmer): 코딩으로 프로그래밍 하는 사람

- **프로그래머가 하는 일:** 코딩으로 프로그래밍하여 프로그램 완성 및 업데이트

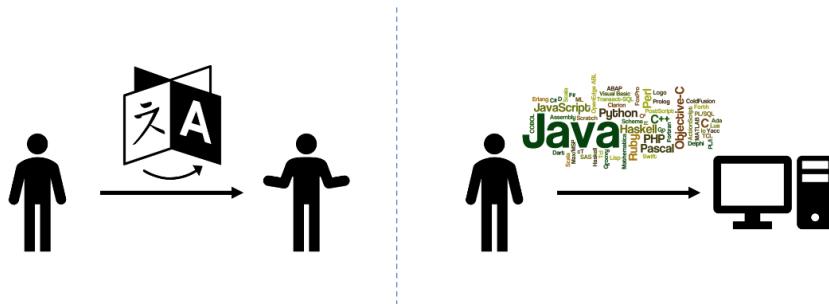
- 세상이 변하기 때문에 프로그램은 한번 만들면 끝이 아니라 **지속적으로 수정**

### 4) 프로그래밍 언어 및 종류: 사람과 컴퓨터가 대화할 수 있도록 연결하기 위해 프로그래머가 사용하는 도구

- **사람언어와 프로그래밍언어:**

- 사람들도 상대가 이해할 수 있는 언어로 대화하듯, 컴퓨터도 컴퓨터가 이해할 수 있는 언어가 있음

- **사람언어(Human Language):** 내가 **다른 사람**과 대화하기 위한 말이나 논리(한국어, 영어, 불어, 스페인어, 철학, 수학, 논리학 등)
- **프로그래밍언어(Programming Language):** 내가 **컴퓨터**와 대화하기 위한 말이나 논리



- **프로그래밍 언어의 종류:** 다양해서 원하는 걸 골라서 쓰면 됩

- Fortran, Cobol, Pascal, C++, Java, Perl, Python, HTML, Ruby, PHP 등

종류	설명
Fortran	과학 계산용 언어로 가장 빠르고 실행 효율성이 높아 군사, 항공, 우주 분야에 많이 쓰임
C, C++, C#	70년대 등장 가장 오랫동안 사용해 온 프로그래밍 언어로 전세계에서 가장 많이 쓰임
Java	91년 오라클이 개발하고 안드로이드 앱 개발하는데 많이 사용
Java Script	웹 브라우저에서 주로 사용하며 페이지를 살아 움직이게 하는데 주로 기여
HTML, CSS	HTML이 웹 페이지의 기능을 주로 만들고 CSS가 이쁘게 디자인을 함
Cobol	기업의 사무용 자료처리로 개발되다 뒤로 밀리고 기계장치에 주로 사용
Pascal	간결하게 작성 가능하고 교육용 언어로 뛰어나다는 평가
R	숫자와 통계정보 처리하는데 특화된 언어
Python	현존하는 가장 쉬운 언어로 사무용, 교육용, 웹개발, 인공지능, 자율주행, IoT, 데이터분석 등에 모두 사용

프로그램 이름	주 생성목적	생성 프로그래밍 언어
휴지통	필요없어서 버린 자료 보관	C, C++, C#
Word	문서 작성	C, C++, C#
한글	문서 작성	C, C++, C#
Excel	수치 계산, 문서 작성, 각종 업무	C, C++, C#
Photoshop	그래픽 이미지 생성 및 처리	C++, C#
Internet, Chrome..	인터넷 환경 접속 및 정보 처리	C++, C#, Java, HTML, CSS
캡처 도구	화면을 이미지로 저장	C++, C#, Java
TeamViewer	원격으로 기기제어	C++, C#, Java, Cobol, Python
카카오톡	커뮤니케이션, 공유	Java, Cobol, Python

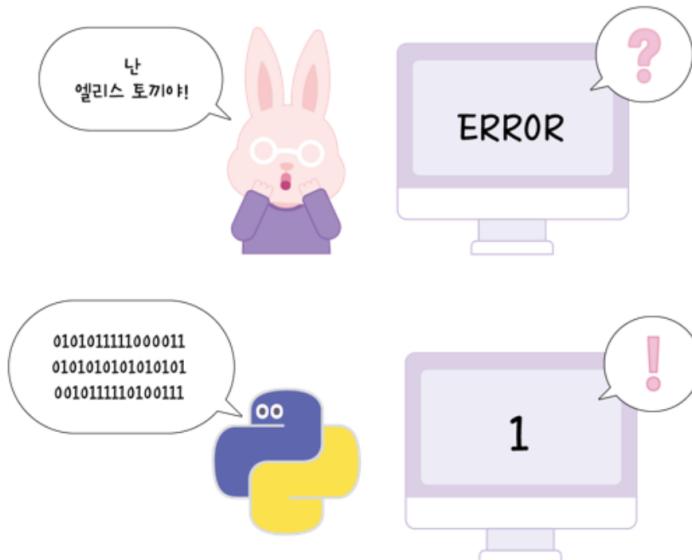
## 5) 프로그래밍 작동원리: 사람과 컴퓨터가 대화할 수 있는 작동원리

- **2진법(Binary System)**: 컴퓨터가 생각하는 방법으로 0과 1만을 사용



- **기계어(Machine Language)**: 0과 1로만 구성된 말

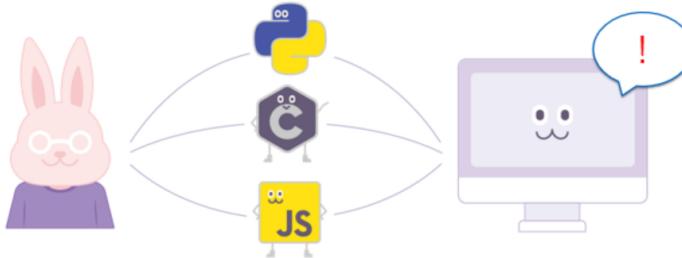
- 컴퓨터는 0과 1로만 생각하니 기계어 만 이해할 수 있음



Copyright @ elice all rights reserved

- **프로그래밍 언어의 등장**: 우리가 프로그래밍 언어로 코드를 작성하면, 특정 프로그램이 기계어로 번역해서 컴퓨터에게 전달

[프로그래밍 언어로 코드 작성]      [기계어로 번역 및 전달]  
`print('난 엘리스 토끼야!')`      ...0100010111011010111...



Copyright @ elice all rights reserved

- **번역기(Compiler/Interpreter)**: 코드를 컴퓨터가 이해할 수 있는 기계어로 번역하는 것
- 프로그래밍을 하기 위해 기계어를 배울 필요는 없음
- 조창기에 기계어로 코딩을 했지만, 프로그램이 복잡해지면서 기계어 코딩이 사실상 불가능해져 대안을 탐색
- 프로그래밍이란 도구를 사람이 쉽게 쓰게 되면서 프로그래밍 산업이 폭발적 증가

## 6) 우리가 해야 하는 것:

"프로그래밍 언어를 배워서 규칙에 맞는 소스 코드를 작성할 줄 알면 됨"

- **코드(Code)**: 컴퓨터에게 말을 걸기 위해 쓴 글
- **소스 코드(Source Code)**: 영어든 한국어든 사용 규칙이 존재하듯, 프로그래밍 언어도 기본 규칙 존재하며 이를 따라 작성된 것

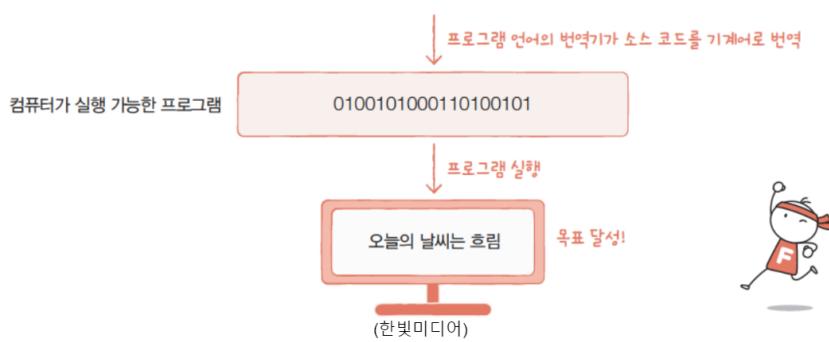
인간의 생각, 의지, 계획

오늘의 날씨를 보여 주는 프로그램을 만들어야지!

↓  
프로그래밍 언어의 문법에 따라 소스 코드 작성

소스 코드

`show_me_the_weather ("today")`



• 요약:

- (1) 컴퓨터와 대화하기 위해서는 컴퓨터가 이해하는 언어, 즉 기계어를 배워야 함
- (2) 기계어는 1과 0으로 모든 것을 표현하는 2진법을 사용
- (3) 기계어는 사람이 이해하기 매우 어렵기 때문에 기계어를 대체할 수 있는 새로운 언어, 즉 프로그래밍 언어 등장
- (4) 프로그래밍 언어의 문법에 따라 사람이 이해할 수 있는 문자로 작성된 프로그램을 소스 코드로 작성
- (5) 프로그래밍 언어가 번역기를 이용해서 소스 코드를 컴퓨터가 이해할 수 있는 기계어로 번역

#### 7) 프로그래밍을 배워야 하는 현실적 이유:

- 우리의 의사와 관계없이 변화는 이미 시작 되었고, 변화의 중심에 기술산업의 혁신이 있고, 이런 변화를 프로그래밍이 리드하며 생존을 위해 지금도 진화중

Rank	Brand	Brand Value	1-Yr Value Change	Brand Revenue	Company Advertising	Industry
#1	Apple	\$205.5 B	12%	\$265.8 B	-	Technology
#2	Google	\$167.7 B	27%	\$136.2 B	\$6.4 B	Technology
#3	Microsoft	\$125.3 B	20%	\$110.2 B	\$1.6 B	Technology
#4	Amazon	\$97 B	37%	\$211.4 B	\$8.2 B	Technology
#5	Facebook	\$88.9 B	-6%	\$48.8 B	\$1.1 B	Technology
#6	Coca-Cola	\$59.2 B	3%	\$23.8 B	\$4.1 B	Beverages
#7	Samsung	\$53.1 B	11%	\$221.6 B	\$3.6 B	Technology
#8	Disney	\$52.2 B	10%	\$33.8 B	\$2.8 B	Leisure
#9	Toyota	\$44.6 B	0%	\$190.8 B	\$4.6 B	Automotive
#10	McDonald's	\$43.8 B	6%	\$96.1 B	\$389 M	Restaurants

- 미래에는 프로그래밍이 선택이 아닌 필수 시대가 될 것이며, 이미 세계 대학들이 필수 교육으로 지정하고, 우리나라도 초등학교에서 수업화 진행중



쉽게 배우는 초등 AI 1  
초등교재

'인공지능 발견하기'를 컨셉으로 한 초등 1~2학년 대상 교재



쉽게 배우는 초등 AI 2  
초등교재

'인공지능 알아가기'를 컨셉으로 한 초등 3~4학년 대상 교재



쉽게 배우는 초등 AI 3  
초등교재

'인공지능 만들기'를 컨셉으로 한 초등 5~6학년 대상 교재



쉽게 배우는 중학 AI

중등교재

'인공지능의 이해와 활용'을 컨셉으로 한 중등 대상 교재



수학과 함께하는 고교 AI 입문

고등교재

체험 중심의 '인공지능 기초'에 수학을 이용하여 인공지능의 다양한 모형의 원리를 조금씩 이해해 보는 고등 대상 교재



수학과 함께하는 AI 기초

고등교재

임상 속 다양한 문제 해결에 인공지능 기술을 적용하기 위한 프로그래밍과 수학과의 연결고리를 풀이한 교재



글자를 읽고 쓰는 법을 배웠다고 모두 전문 작가가 되어야 할까요?



요리를 배웠다고 모두 전문 요리사가 되어야 할까요?

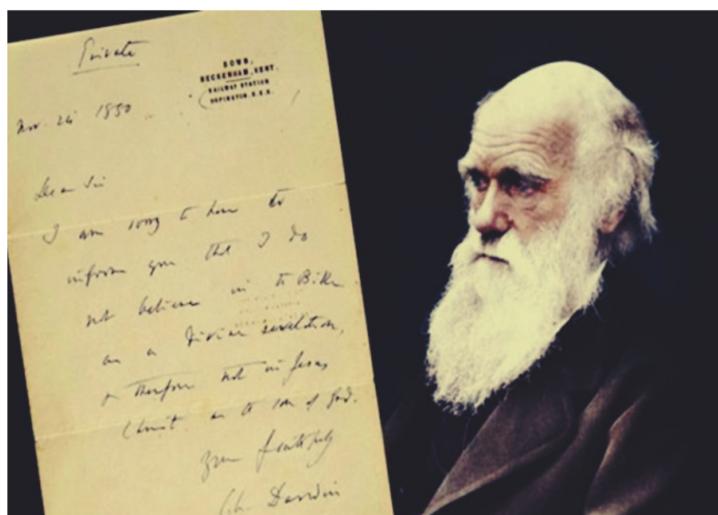


프로그래밍을 배웠다고 모두 프로그래머가 되어야 할까요?

- 일상의 제품과 서비스가 프로그램으로 이루어진 만큼 프로그래밍으로 세상을 조금 더 빨리 이해 할 수 있을 것

가장 강한 종이 살아는 것이 아니다.  
가장 두뇌가 뛰어난 종이 살아남는 것도 아니다.  
단지 변화에 잘 적응하는 종이 살아남는다.

- 찰스 다윈 -



## 2.2 프로그래밍 기반 데이터분석

"비즈니스 작동요소 = 조직 작동방식 = 사람 교류방식 = 컴퓨터 교류방식 = 데이터과학 작동방식"

1) 사람 교류방식 vs 컴퓨터 교류방식:

4) 젠장(검증)



1) 밥먹자(입력)  
3) 좋아( 출력)

2) 어쩌지(처리)



4) 좋아(검증)



1) 밥먹자(입력)  
3) ... ( 출력)

2) ... (처리)



2) 실제 교류방식: 반복적으로 피드백을 업데이트 하며 관계를 업데이트

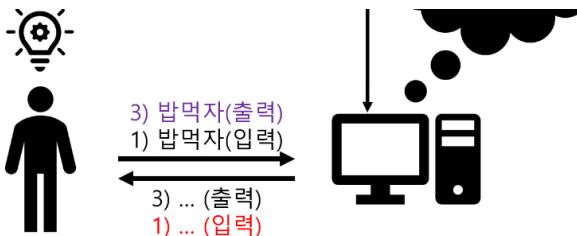
2) 일정잡아볼까?(처리)  
1) 언제먹을래(입력)  
3) 언제먹을래( 출력)

4) 좋아(검증): 3) vs 3

4) 머지?(검증): 3) vs 3



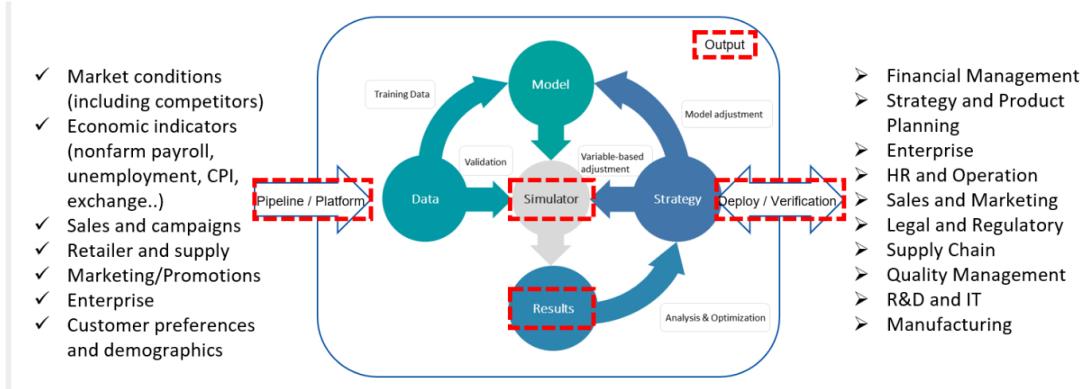
2) ... (처리)



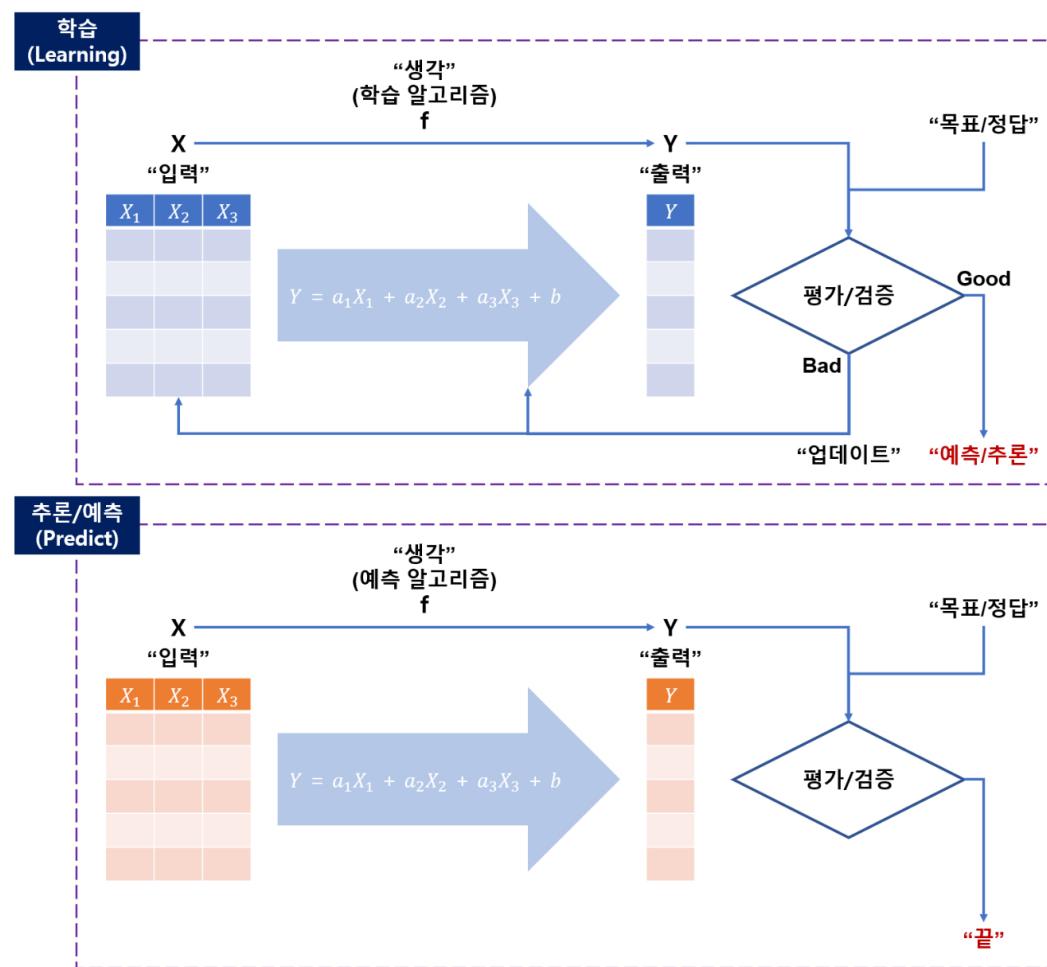
- 1) 컴퓨터가 좋다는데?(입력)  
2) 친하게 지내볼까?(처리)

3) 데이터과학 작동방식: 입력(x)/처리(f)/출력(y)/검증(e)

- 이상적:



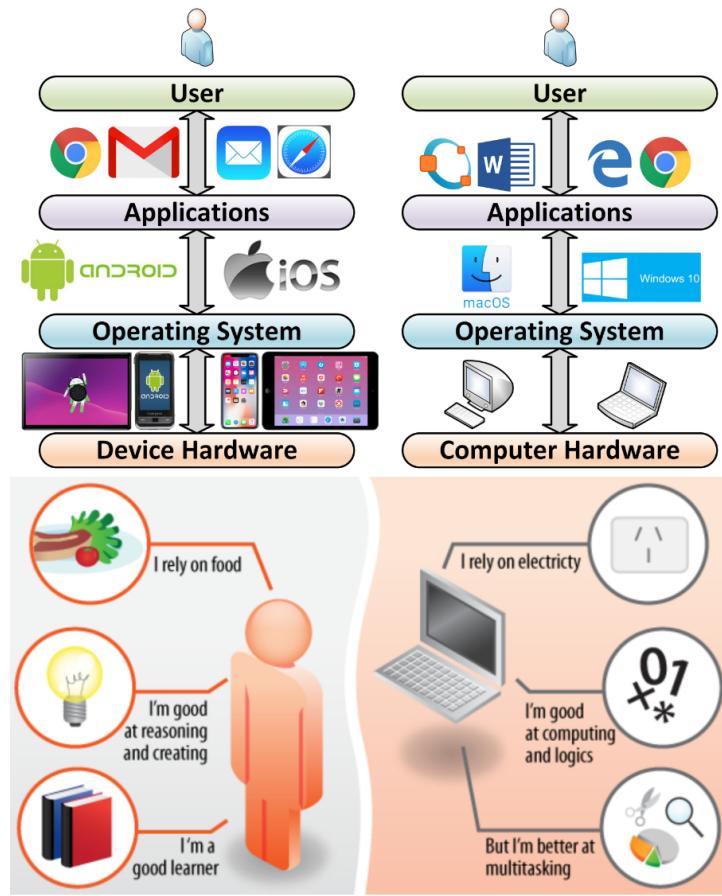
- 현실적: 끊없는(?) 학습과 예측의 무한반복 기반, 끝이 줄 알았는데 반쪽짜리



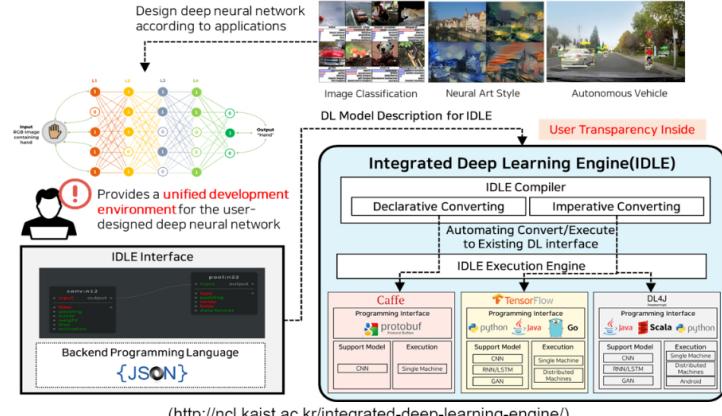
- 4) 결론:

프로그래밍 언어를 사용하여,

(1) 프로그래밍 된 컴퓨터와 기계들을 손쉽게 다룰 수 있고, 인간의 단점을 극복 할 수 있는 강력한 도구 획득

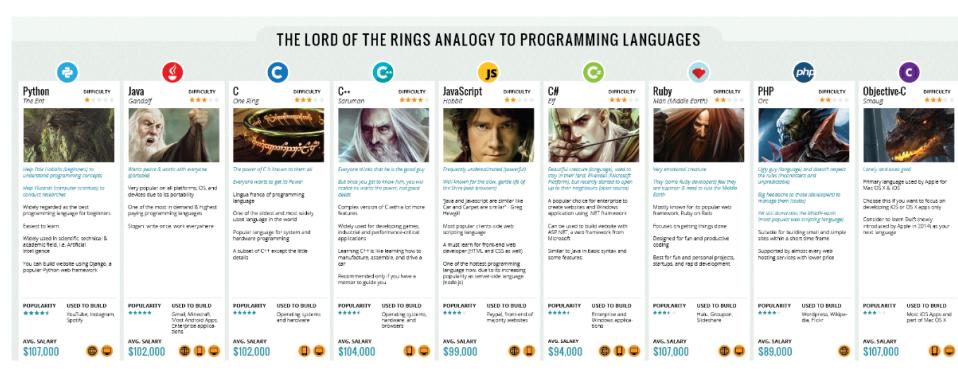


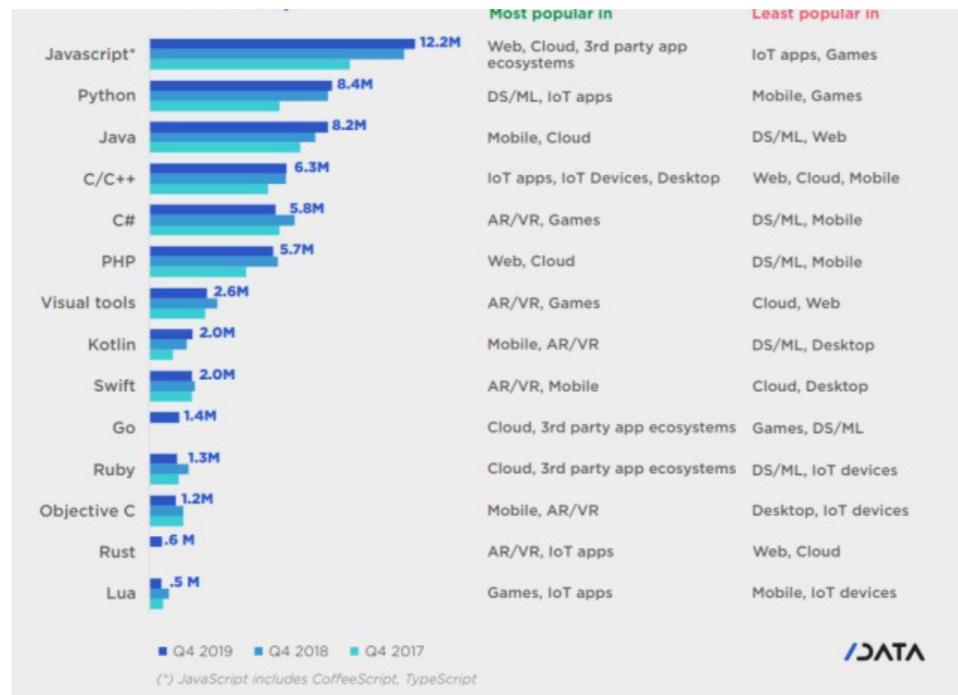
(2) 데이터 비즈니스, 빅데이터처리, 인공지능 등이 가능해진 프로그래밍 기반 기계의 진화 덕분에 인간도 함께 진화 할 수 있는 기회가 생겼으며 선택 여부는 인간의 몫!



## 2.3 데이터분석 특화 프로그래밍 언어 파이썬(Python)

### 1) Python:





- 네덜란드의 귀도 반 로섬(Guido van Rossum)이 개발한 프로그래밍 언어(89년 취미로 제작 + 91년 외부 공개)

- 세계적인 기업들(구글, 아마존 등)이 필수도구로 사용하고 성능이 입증된 언어

- 인공지능(머신러닝, 딥러닝)을 빠르게 배우고 활용 할 수 있는 컴퓨터 언어

- 데이터 분석과 머신러닝 특화 많은 세부도구(Library)를 포함하여 확장성과 범용성이 높음

▪ **라이브러리/패키지:** 특정 기능을 자동으로 실행하도록 미리 만들어 놓은 것

- \*\*누구나 활용 가능한 오픈소스\*\*로 공개되어 접근성과 활용성, 그리고 확장성이 용이(ex. R, SPSS, etc.)
- 간결하고 쉬운 컴퓨터와의 소통언어로 쉽게 배울 수 있음

소스 코드 1 - C언어	#include <stdio.h> int main() { printf("Hello, World!"); return 0; }
소스 코드 2 - 자바	public class HelloWorld { public static void main(String[] args) { System.out.println("Hello, World!"); } }
소스 코드 3 - 파이썬	print("Hello, World!")

```
if x in [1,2,3,4]:    # '만약 x가 [1,2,3,4] 중에 있다면?' 파이썬을 본적 없지만 뜻은 이해가됨!
```

### 1) Anaconda: Python기반의 Open Data Science Platform

- **Data Science Library:** 데이터 사이언스와 머신 러닝 분야에서 파이썬을 사용하기 위해 기본적으로 설치하는 배포판

- (a) Jupyter 같은 IDE(Integrated Development Environment) 개발도구
- (b) Pandas, Numpy, Matplotlib 등 데이터분석에 유용한 도구(Library) 포함
- (c) Numpy, SciPy 같은 과학 분석용 라이브러리
- (d) Matplotlib 같은 데이터 시각화(Data Visualization) 라이브러리
- (e) TensorFlow 같은 머신 러닝(Machine Learning) 라이브러리



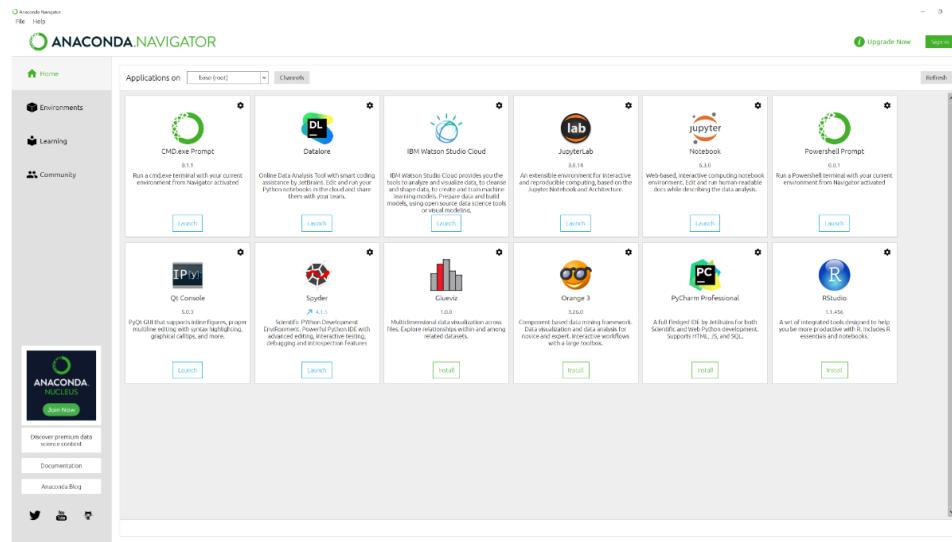


## CONDA

Data Science Package & Environment Manager

- Python을 포함하여 Python Library 등을 하나로 정리해 둔 배포판
- 추가적인 Library를 연결 및 확장하여 개발/웹/오피스/비즈니스로 활용 가능
  - 어떤 라이브러리를 써야할지 잘 모르겠을때: <https://github.com/vinta/awesome-python>
  - 주목받는 파이썬 프로젝트들을 둘러보고 싶을때: <https://github.com/trending/python>

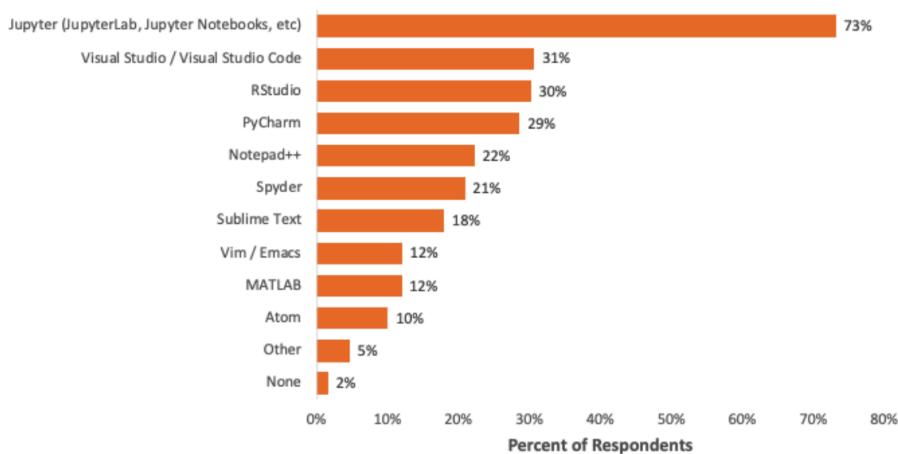
- **Navigator:** 라이브러리 설치나 외부 IDE 개발 도구를 쉽게 연결하도록 한 포털 기능



### 2) IDEs: Jupyter Notebook / Jupyter Lab / PyCharm / Spyder / RStudio ...

- Interactive 환경을 인간에게 제공하여 컴퓨터(Programming Platform)와 소통을 더욱 쉽게함
- Anaconda 프로그래밍 환경을 그대로 사용
- 코딩하면서 바로 결과를 확인할 수 있음
- 문서화(Markdown)와 클라우드(Git, Google Drive 등) 연결/저장 등이 가능
- 각종 단축키와 투터리얼 등의 자료가 많음

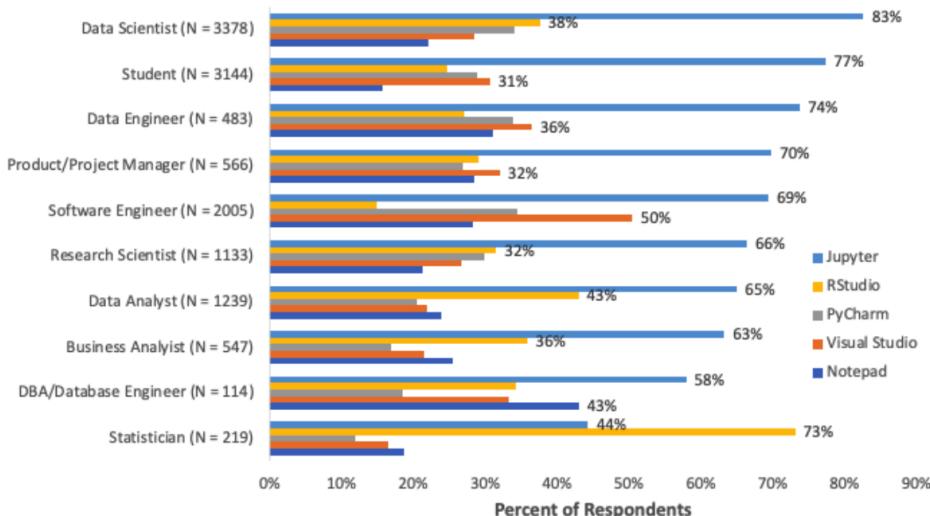
### Which of the following integrated development environments (IDEs) do you use on a regular basis?



Note: Data are from the 2019 Kaggle ML and Data Science Survey. You can learn more about the study here: <https://www.kaggle.com/c/kaggle-survey-2019>.

A total of 19717 respondents completed the survey; the percentages in the graph are based on a total of 14762 respondents who were asked and who answered the question.

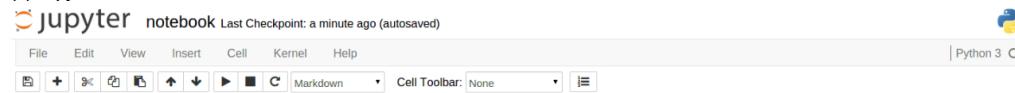
## Use by Job Title



Note: Data are from the 2019 Kaggle ML and Data Science Survey. You can learn more about the study here: <https://www.kaggle.com/c/kaggle-survey-2019>. A total of 19717 respondents completed the survey; the percentages in the graph are based on a total of 14762 respondents who answered the question, "Which of the following integrated development environments (IDEs) do you use on a regular basis?"

- 예시:

### (1) Jupyter Notebook



#### Simple spectral analysis

An illustration of the [Discrete Fourier Transform](#) using windowing, to reveal the frequency content of a sound signal.

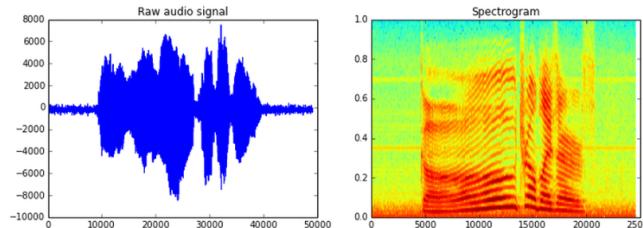
$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi}{N} kn} \quad k = 0, \dots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

```
In [1]: from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin specgram routine:

```
In [2]: %matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title("Raw audio signal")
ax2.specgram(x); ax2.set_title("Spectrogram");
```



### (2) Jupyter Lab

In this Notebook we explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= px - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

```
In [4]: from lorenz import solve_lorenz
t, x_t = solve_lorenz(N=10)
```

Output View

lorenz.py

```

9 def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
10     """Plot a solution to the Lorenz differential equations."""
11     fig = plt.figure()
12     ax = fig.add_axes([0, 0, 1, 1], projection='3d')
13     ax.axis('off')
14
15     # prepare the axes limits
16     ax.set_xlim((-25, 25))
17     ax.set_ylim((-35, 35))
18     ax.set_zlim((5, 55))
19
20     def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
21         """Compute the time-derivative of a Lorenz system."""

```

```

x, y, z = x_y_z
return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]

# Choose random starting points, uniformly distributed from -15 to 15
np.random.seed(1)
x0 = -15 + 38 * np.random.random((N, 3))

```

### (3) PyCharm

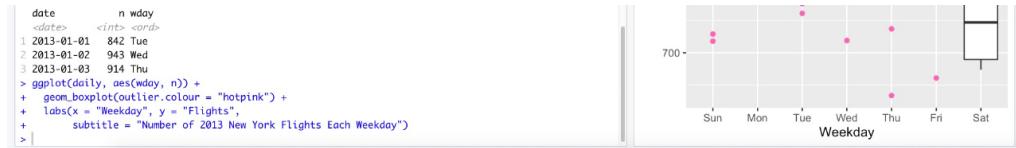
PyCharm interface showing a project structure with files like `tests.py`, `models.py`, and `admin.py`. A search bar is open over the code editor. Below it, a database browser shows the `Django default` database with tables like `auth_group`, `auth_permission`, and `auth_user`. A debugger window is open, showing a stack trace for a test case named `test_index_view_with_a_future_question`. The test case uses `assertEqual` to check if the response context contains the latest question list. The code editor shows imports for `reverse`, `assertQuerysetEqual`, and `assertContains`.

### (4) Spyder

Spyder interface showing a project explorer with various Python files and modules. A variable explorer window is open, listing variables like `array_int8`, `array_uint32`, `bars`, `df`, `filename`, `list_test`, `nrows`, `r`, `radial`, `region`, `rgb`, and `series`. A 3D surface plot is displayed in the main workspace, showing a complex landscape. A 2D polar plot is also visible.

### (5) RStudio

RStudio interface showing an R script titled `flights-example.R`. The script loads packages like `lubridate`, `dplyr`, and `ggplot2`, and performs data manipulation on the `flights` dataset. The console shows the resulting data frame. To the right, an environment browser shows objects like `daily` and `flights`. A boxplot titled "Number of 2013 New York Flights Each Weekday" is displayed, showing the distribution of flights per day of the week.



## (6) Visual Studio

