

스마트 모빌리티 프로그래밍

Ch 1. 스마트 모빌리티 프로그램 개요, 파이썬 프로그램의 기본 구조



영남대학교 정보통신공학과
교수 김 영 탁

(Tel : +82-53-810-2497; E-mail : ytkim@yu.ac.kr)

Outline

- ◆ 스마트 모빌리티
- ◆ 스마트 모빌리티의 핵심 기술
- ◆ 컴퓨팅 사고
- ◆ 컴퓨터에서의 정보 표현
- ◆ 파이썬 프로그래밍 준비
- ◆ 파이썬 프로그램의 기본 구성
- ◆ 파이썬 프로그램의 식별자 (identifier), 기본 연산
- ◆ 파이썬 프로그램의 입력과 출력
- ◆ 파이썬 프로그램의 실행 제어 기초 – 조건문과 반복문
- ◆ 객체지향형 프로그래밍 개요 (기본 용어)
- ◆ 터틀 그래픽 개요 – 도형 (사각형, 삼각형, 별 (star)) 그리기



스마트 모빌리티 (Smart Mobility)

스마트 모빌리티 (Smart Mobility)

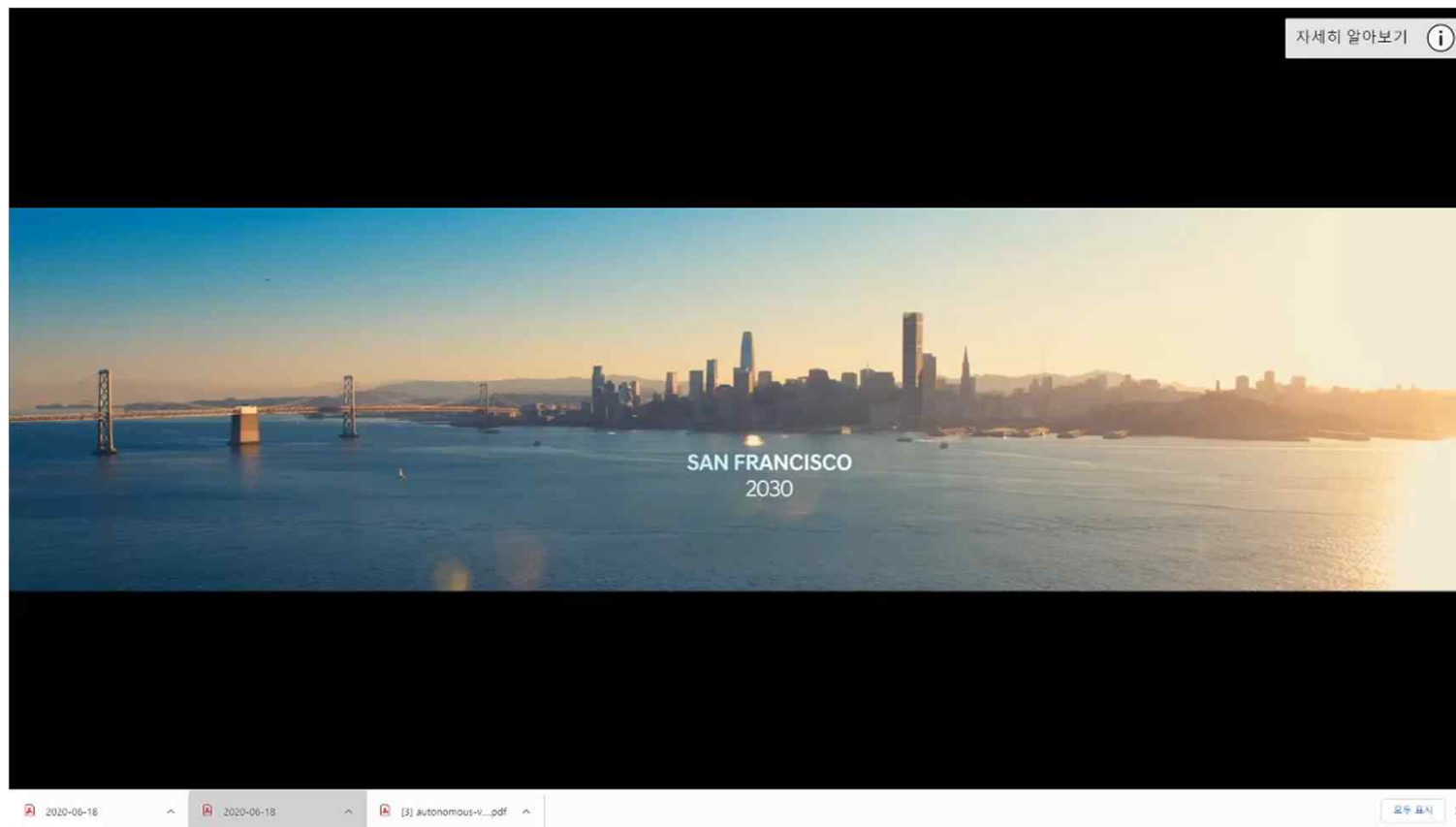
◆ 스마트 모빌리티란 ?

- 첨단 정보통신기술 (ICT)과 차량/비행 기술이 융합되어 효율적이며 편리한 미래 교통 및 물류 수단을 포괄적으로 의미
- 현재 사용중인 정보통신기술 (ICT)과 개발중인 상호 연결성/협력 기반의 첨단 정보통신기술을 육상 교통, 항공, 선박, 물류 등에 융합하여 사람/물건의 이동을 편리하고, 안전하며, 효율적으로 처리할 수 있게 함
- 지역적인 이동 및 광역 이동을 포괄적으로 지원
- 도시 지역에서의 사람/물건 이동성을 획기적으로 개선
- 이동 수단의 공유 (shared mobility)와 전기/수소 차량/드론의 군집 주행/비행을 함께 고려하며, 에너지 효율적인 교통/물류 체계를 제공
- 도시지역 주민의 삶의 질이 향상될 수 있게 하며, 육상 교통, 항공, 선박 등의 교통/물류 체계에 수준 높은 이동성을 지원

Example Smart Mobility

◆ Example Smart Mobility Solution by Hyundai (proposed in CES2020)

(Source: https://www.youtube.com/watch?v=b4nQWG5z_z8)



Example Smart Mobility (Proposed by Hyundai in CES2020)

◆ Example Smart Mobility Solution by Hyundai

- UAM (Urban Air Mobility), Personal Air-Vehicle
- Hub (Sky Port)
- PBV (Purpose Built Vehicle)



(Source: https://www.youtube.com/watch?v=b4nQWG5z_z8)

Google Waymo

◆ Google Waymo Autonomous Car

- successful safe self-driving up to 20 million miles (32,000,000 Km)
- Develops full-auto self-driving car with Fiat Chrysler Automobiles (FCA)
- Establish production line in Detroit for Level 4 self-driving car
- Launches an actual public driverless taxi service in the Phoenix area



(Source: <https://arstechnica.com/cars/2020/10/waymo-finally-launches-an-actual-public-driverless-taxi-service/>)

Tesla Self Driving

◆ Tesla's features making autopilot possible

- A forward-facing radar
- Cameras provide visibility up to 250 meters away
- A high-precision digitally-controlled electric assist braking system
- 12 long-range ultrasonic sensors around the car
- sensors can be affected if there is debris covering them
- Sense everything within 16 feet away from the vehicle
- *Tesla is not using LiDAR !*

(Source:

- <https://www.jameco.com/Jameco/workshop/HowItWorks/how-it-works-tesla-autopilot-self-driving-automobile-technology.html>
- Tesla full self driving BETA, <https://www.youtube.com/watch?v=qEejCZZAQc>,
- Tesla FSD Beta - 17 minute drive with ZERO disengagements!, https://www.youtube.com/watch?v=mYpA_US_RzM

)



Urban Air Mobility (UAM)

◆ Ehang Air Mobility

- Passenger Transportation
- E-port and intelligent parking



(Source:

- https://www.youtube.com/watch?v=XTJDkSmDf_s
- <https://www.ehang.com/uam/>
- <https://www.youtube.com/watch?v=d66MoI4GdFs>
- <https://www.youtube.com/watch?v=ugkVrrr8I-4>)



Hydrogen Fuel Cell Drone

◆ Hydrogen Fuel Cell for Drone by Doosan mobility

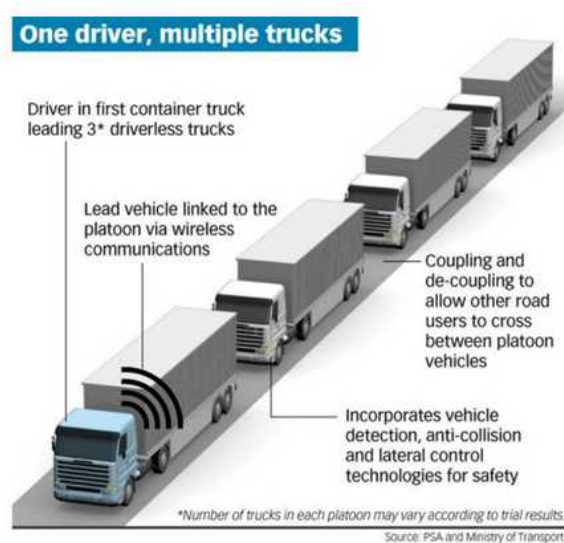
- extended flight time : up to 2 hours
- Technology innovation award in CES2020



(Source: https://www.youtube.com/watch?v=KtxS5dW_YNQ;
<https://www.youtube.com/watch?v=A9rjEASZiz0>)

Platooning of Vehicles

◆ One Driver, Multiple Trucks



(Source:

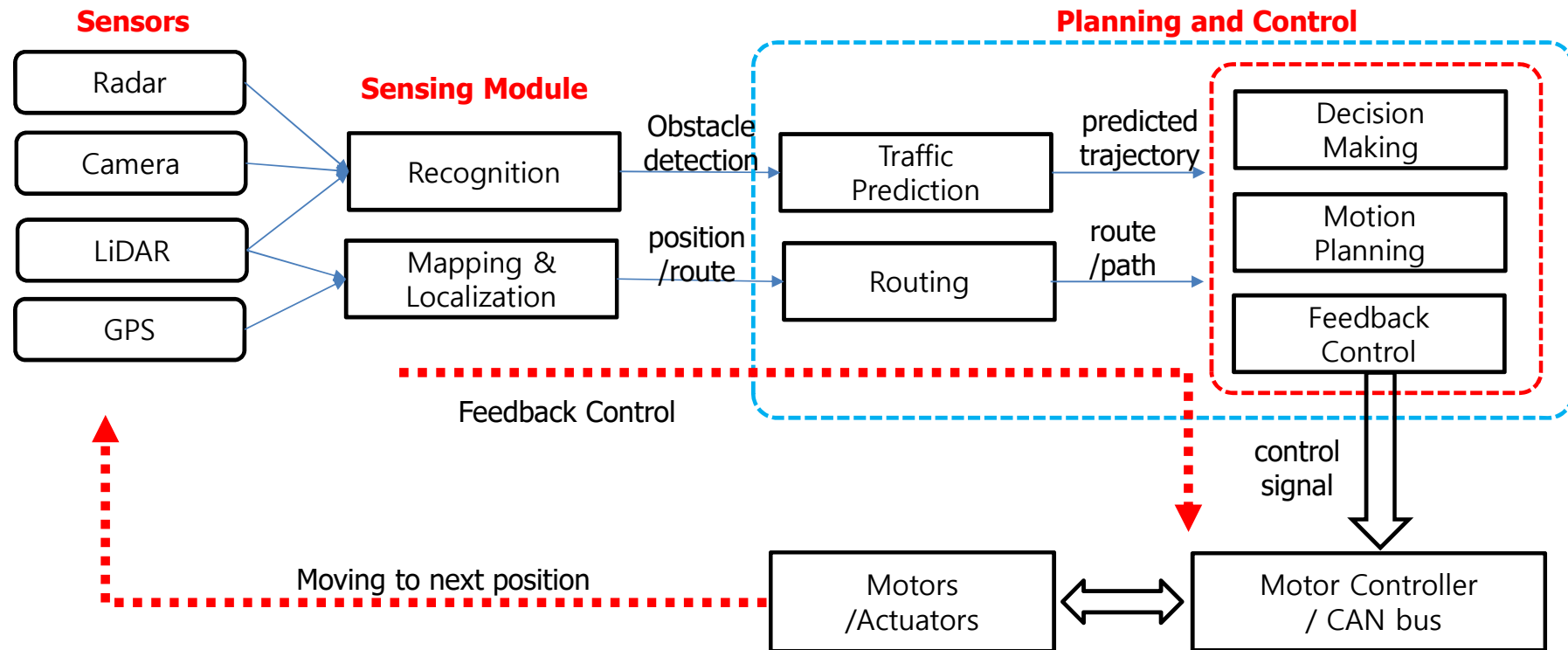
- https://www.lotis.or.kr/tia03v.do?cn_id=2018101500012&cn_type_cd=TIA&CN
- Scania, Semi-autonomous truck platooning — how does it work?, <https://www.youtube.com/watch?v=lpwG4A56r0>
- Volvo self-driving truck platoon in the European Truck Platooning Challenge, <https://www.youtube.com/watch?v=lx9EFJ6ggZc>
- Vision Truck Platooning 2025: Creating Next Generation Mobility, <https://www.youtube.com/watch?v=I-xMdybBzUY>

)

ch 1- 11



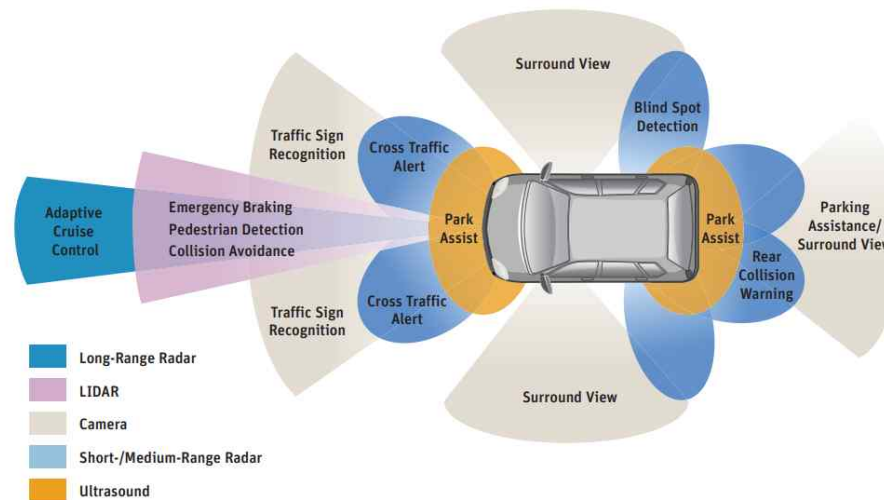
Feedback Control in Autonomous Driving



Radar

◆ Radio Detection and Ranging (Radar)

- Obstacle detection using radio frequency (RF) signal transmission/reflection/reception
- Ranging by ToF (time of flight)
- Measuring relative speed of obstacle



(Source:

- <https://www.ansys.com/-/media/ansys/corporate/resourcelibrary/article/autonomous-vehicle-radar-aa-v12-i1.pdf>

- General Radar, 4D,

https://www.genrad.io/?qclid=Cj0KCQjwufn8BRCwARIsAKzP6951YmMMFMcGOjKjT_rtlE_4uoT_a5O7u6kt7w4rhfgCTiUpSZGWoaAIwrEALw_wcB

)



LiDAR

◆ LiDAR (Light Detection and Ranging)

- Measure time of flight (ToF)
- Short ranging and 3D image mapping

◆ Future

- LiDAR installed inside vehicle
- Head lamp with LiDAR

◆ Radar vs. LiDAR

	Radar	LiDAR
Frequency	Long wave (low frequency)	Short wave length (high frequency)
Merit	Longer detection/ranging Chip	Short ranging Higher resolution 3D map
Demerit	Lower resolution	expensive

(Source:

- 2020 Apple iPad Pro LiDAR Scanner, https://www.youtube.com/watch?v=fS3J4V_BgP0,
- Compact UAV LiDAR Scanning System, <https://www.youtube.com/watch?v=gjxgFowijh8>
- Hovermap - World's first autonomous LiDAR mapping payload, <https://www.youtube.com/watch?v=2zadTtCadeI>
- Google Waymo,

)



2D, 3D Camera

◆ Current Status

- Object (road sign, obstacle,) image recognition
- Single (mono) lens

◆ Future

- Stereo lens
- 3D cameras in autonomous vehicles
- more precise information by combining cameras

(Source:

- <https://www.bosch.com/stories/mpc3-self-driving-car-camera/>

- 3D Cameras in autonomous vehicles, <https://future-markets-magazine.com/en/markets-technology-en/3d-cameras-in-autonomous-vehicles/>

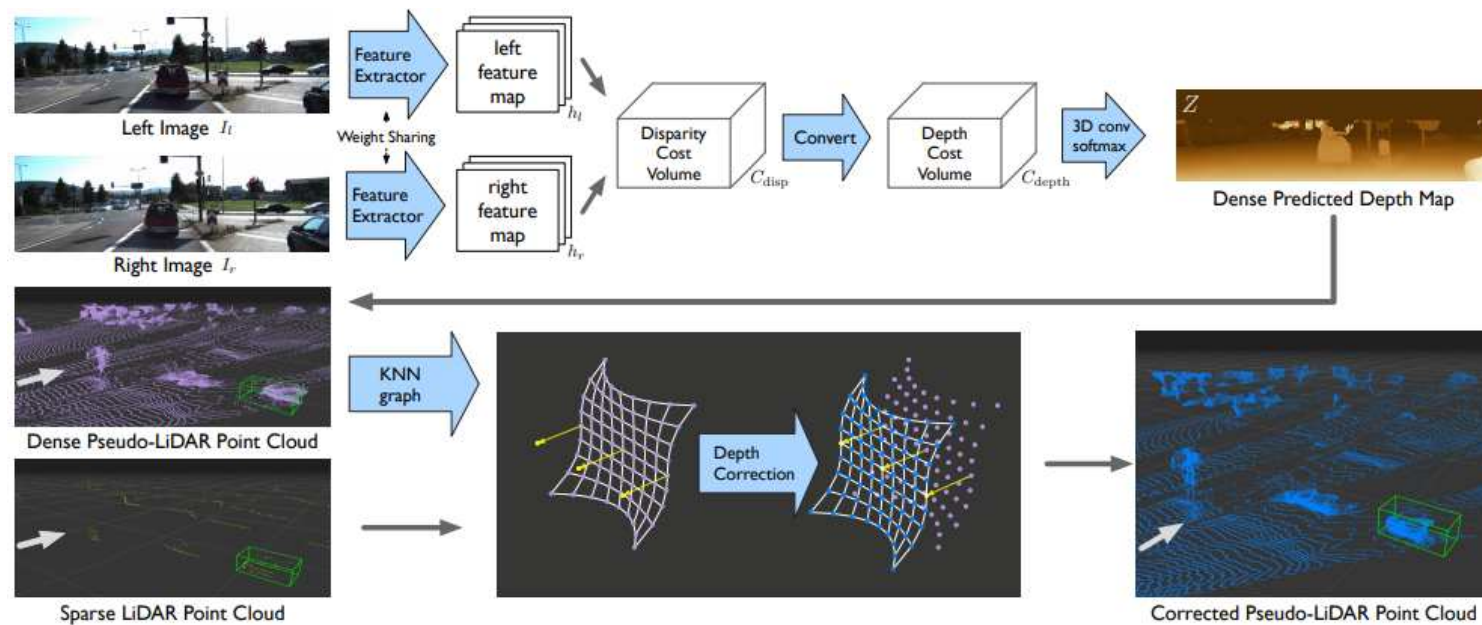
)



Pseudo LiDAR

◆ Pseudo-LiDAR++

- accurate depth for 3D object detection in autonomous driving
- Published at ICLR 2020 by Yan Wang et. al, Cornell Univ.
- source: <https://arxiv.org/pdf/1906.06310.pdf>



Global Positioning System(GPS)

◆ Current GPS Services

- Positioning based on ranging from GPS satellites
- Resolution: currently in meter unit order
- GPS by USA
- Beidou by China
- GLONASS (Global Navigation Satellite System) by Russia
- Quasi-Zenith Satellite System (QZSS) by Japan
- K-GPS by Korea in 2022

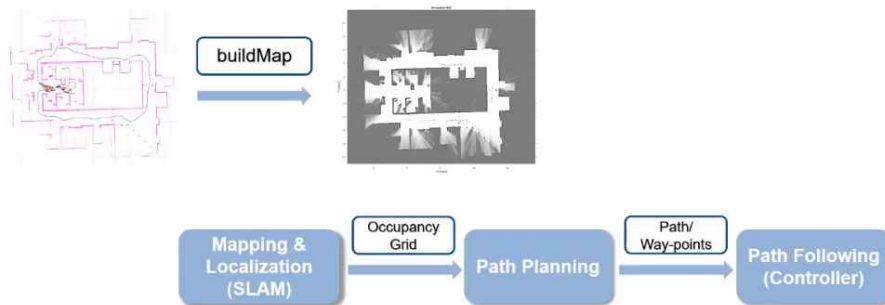
◆ Future

- Higher resolution GPS in cm unit order

SLAM

◆ SLAM (Simultaneous Localization and Mapping)

- the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it
- Popular approximate solution methods include the particle filter, extended Kalman filter, Covariance intersection, and [GraphSLAM](#)
- SLAM algorithms are used in navigation, robotic mapping, and [odometry](#) for [virtual reality](#) or [augmented reality](#)



(Source:

- How does SLAM work? https://www.youtube.com/watch?v=IH_n9bfy-nM
- OpenSLAM, https://www.youtube.com/watch?v=Ro_s3Lbx5ms
- SLAM with MatLab, <https://www.youtube.com/watch?v=XZxpmS0QuHI>

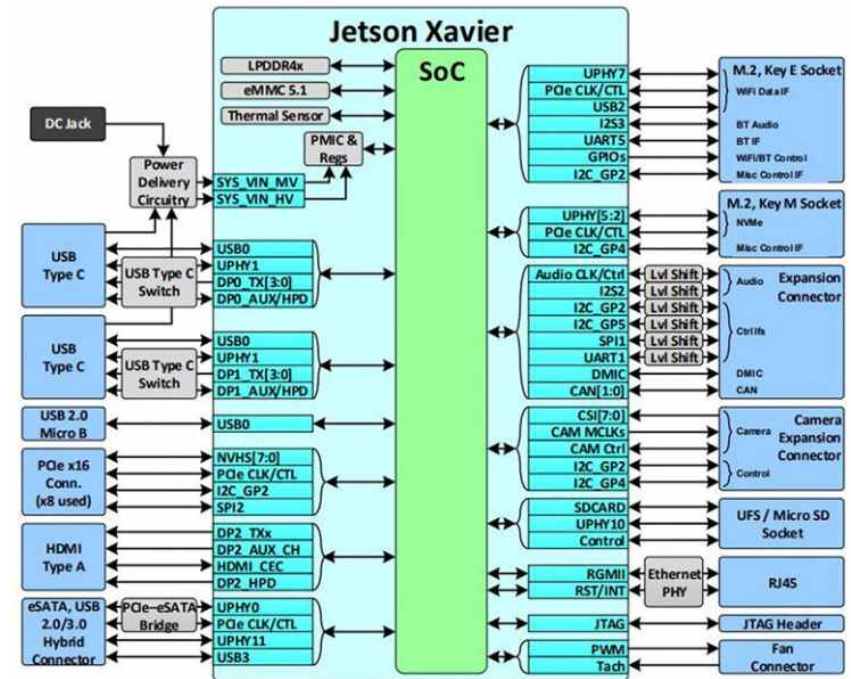
)



AI and Machine Learning on Vehicles

◆ NVIDIA GPU Platform

- open AV development platform for software developers
- includes DRIVE software plug-ins that allow partners to integrate their technology with the DRIVE AGX
- DRIVE AGX platform is an in-Vehicle AI computer that supports autonomous driving with processing of data from camera, radar, and LiDAR sensors to perceive the surrounding environment, localize the car to a map, and plan and execute a safe path forward

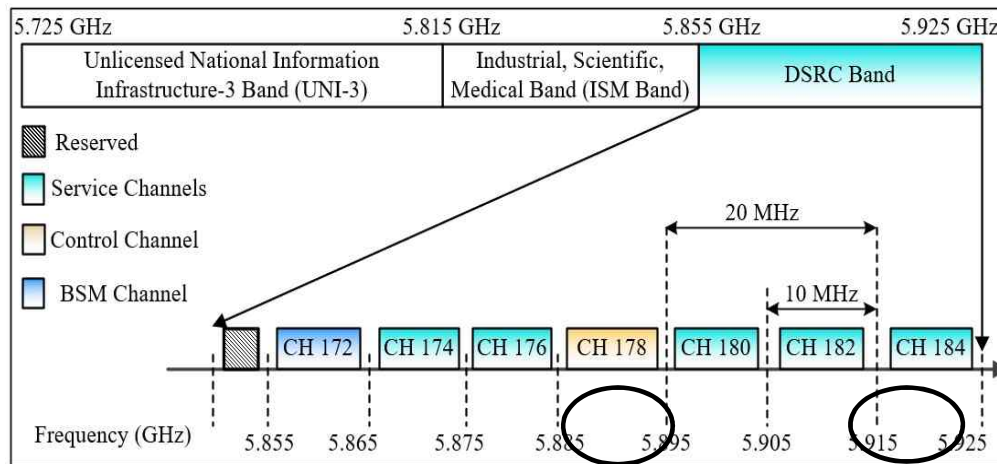


(Source: <https://developer.nvidia.com/embedded/jetson-agx-xavier-developer-kit>)

Vehicular Networking with WAVE / IEEE 802.11p

◆ DSRC/WAVE

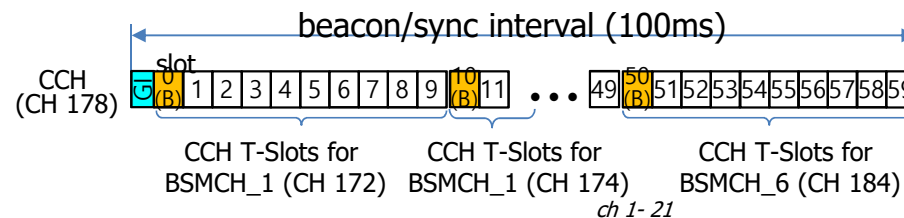
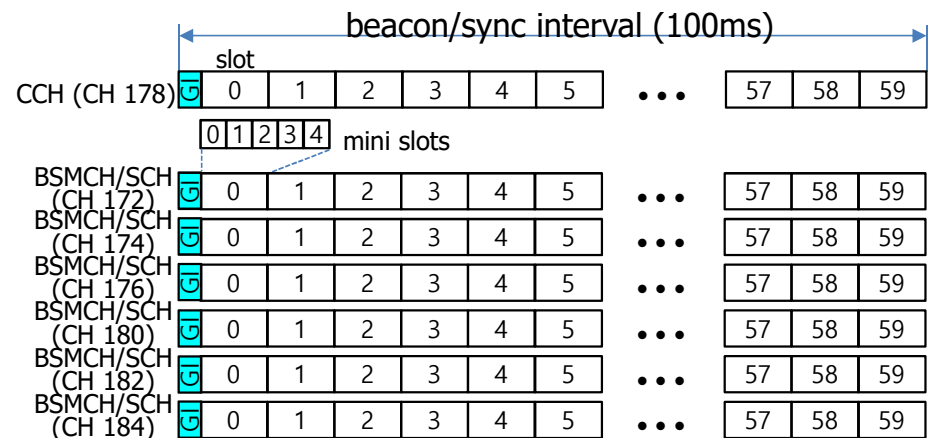
- The U.S. Department of Transportation (DOT) and the Crash Avoidance Metrics Partnership (CAMP) Vehicle Safety Communications 2 (VSC2) Consortium have proposed a new deployment option –
 - CH 178 is left for the exchange of management information only (as control channel), including *WAVE Service Advertisements (WSAs)*
 - CH 172 is dedicated to V2V *Basic Safety Messages (BSMs)*





WAVE / IEEE 802.11p with Slotted TDMA Multi-Channel MAC

◆ Slotted TDMA Multi-Channel MAC (STMC-MAC)

- Beacon Interval (100 ms) is divided into 60 TDMA slot; each TDMA slot is further divided into 5 mini-slots
- CCH and 6 BSMCHs contain total 300 mini-slots; each mini-slot delivers BSM message
- One CCH is shared by overlapped Vehicular Network Cells

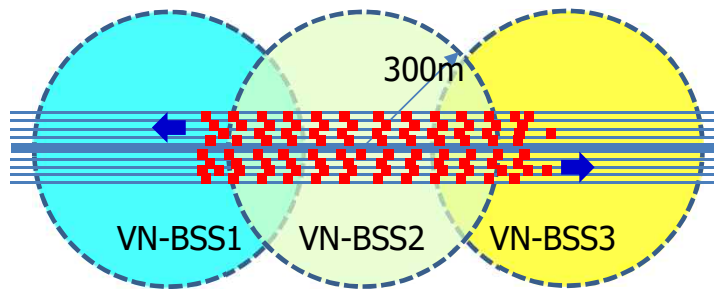


 Guard Interval
 Beacon Frame/Slot

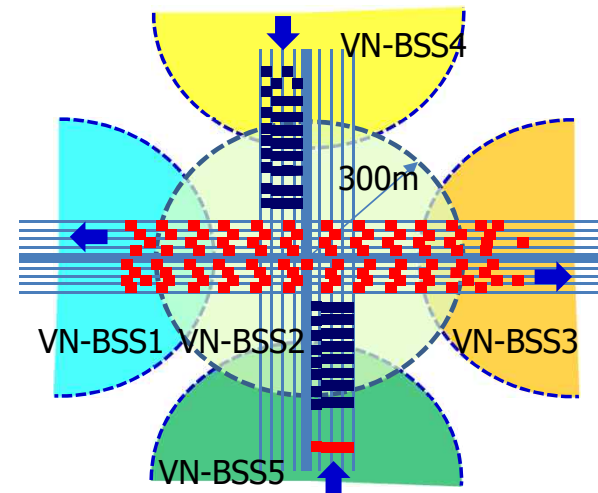
Overlapped Vehicular Network

◆ Overlapped Vehicular Networks in Metropolitan Area

- BSMCH is allocated for each VN-BSS (Vehicular Network Basic Service Set)
- Multiple VN-BSSs are overlapped in linear or intersection topology
- one CCH is shared by multiply VN-BSS
- newly arrived vehicle registers using CCH
- in overlapped VN-BSS region, vehicles broadcast BSM using both BSMCH and CCH to correctly deliver BSM to its neighbor in different VN-BSS (using different BSMCH)
- by using smart handover, the overhead of re-registration at each VN-BSS is minimized



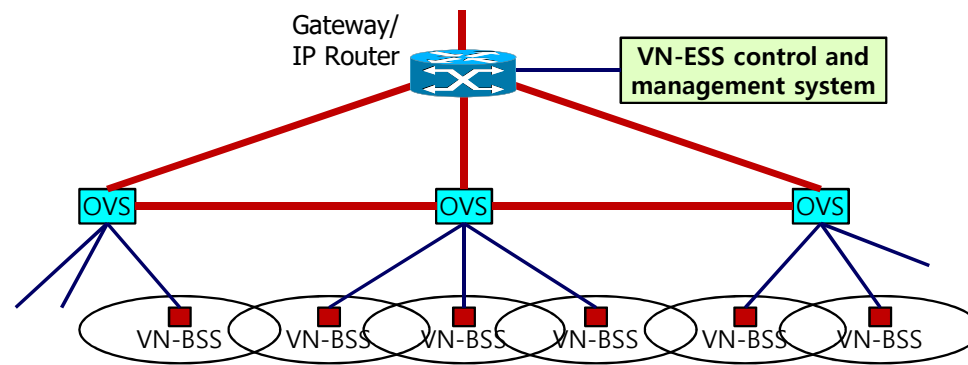
(a) linear (high-way) topology



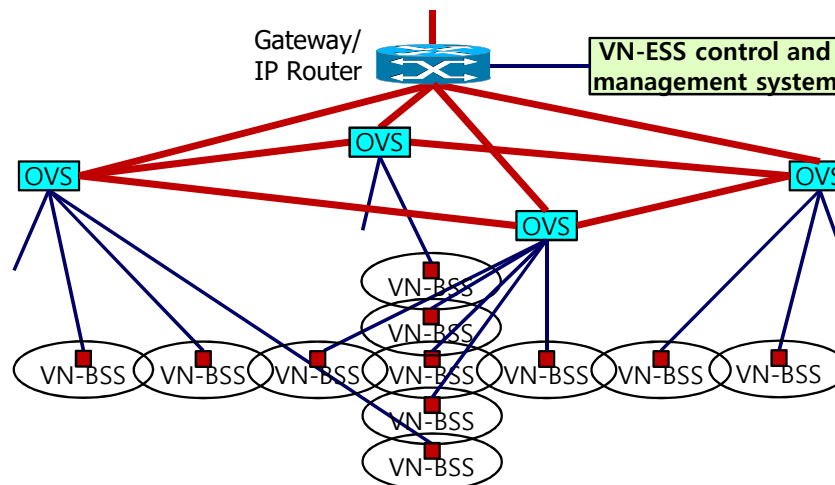
(b) intersection topology

SDN-based VN-ESS

◆ Overlapped Vehicular Network with SDN (Software Defined Networking)



VN-BSS: Vehicular Network Basic Service Set
VN-ESS: Vehicular Network Extended Service Set
OVS: Open vSwitch

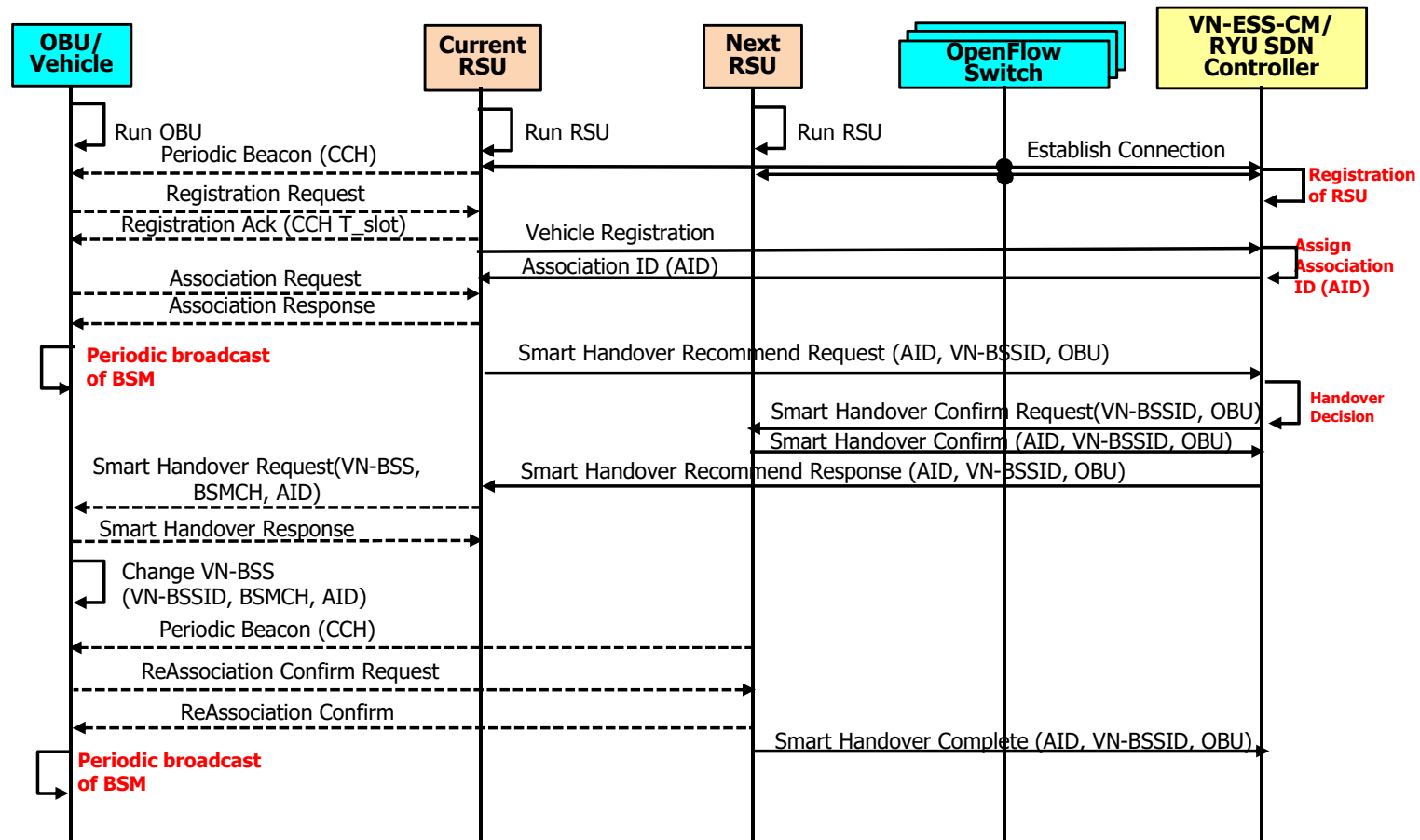


ch 1- 23



Network-initiated Smart Handover

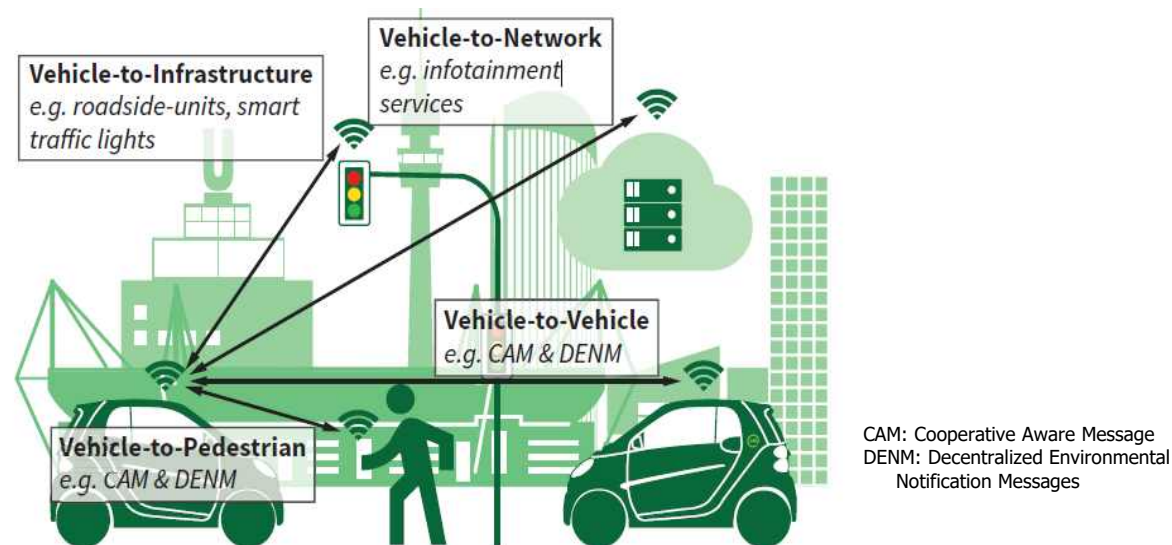
◆ Smart Handovers in Overlapped Vehicular Networks



Vehicular Networking with C-V2X Mode 3/4 Communication

◆ C-V2X(Cellular Vehicle to Everything)

- The 3GPP(Third Generation Partnership Project) cellular vehicle to everything(C-V2X) standard was released in 2017(rel. 14).
- The 3GPP specifies four types of C-V2X applications.



C-V2X

◆ C-V2X

- LTE-V2V adopts single carrier frequency division multiple access (SC-FDMA) at PHY and MAC layers
- resource blocks (RBs) are allocated in pairs, corresponding to 180KHz bandwidth (12 subcarriers with 15KHz space) and 1 ms duration (14 OFDM symbols, of which 9 carry data, 4 are used for channel estimation, and one for timing and possible tx-rx switch)
- minimum allocation time interval is 1 ms; transmission time interval (TTI)
- resource blocks (RBs) are also grouped in the frequency domain into sub-channels, which are all of a given size
- each sub-channel can carry at most one data packet, although one data packet can span over more than one sub-channel
- Data is transmitted in Transport Blocks (TBs)
- a TB contains a full packet to be transmitted, e.g., a beacon or a CAM (cooperative awareness message)/BSM(basic safety message)
- each TB is transmitted with a side-link control information (SCI) that occupies 2 RBs in the same sub-frame, and represents the signaling overhead in C-V2X Mode 4
- the SCI includes information such as the modulation and coding scheme used to transmit the TB, and the RBs used to transmit the TB

스마트 모빌리티의 요약 정리

◆ Current Status of Smart Mobility

- connected vehicles, automotive vehicles, and urban air mobility(UAM) are hot topics now and in near future
- Sensors (GPS, Radar, LiDAR, Camera) for object/obstacle recognition, mapping and localization (SLAM)
- WAVE/IEEE 802.11p and C-V2X are under development
- commercial self-driving taxi service is available now

◆ Future Works

- enhanced sensors (Stereo camera, LiDAR, Radar) and enhanced SLAM are essential for upgraded safety
- better SLAM with less-expensive sensors (e.g., Pseudo LiDAR)
- enhanced vehicular networking technologies are essential for upgraded safety
- infrastructures for smart shared mobility must be developed

컴퓨팅 사고 (Computational Thinking)
프로그래밍 언어 (Programming Language)

컴퓨팅 사고

◆ 컴퓨팅사고 (Computational Thinking)

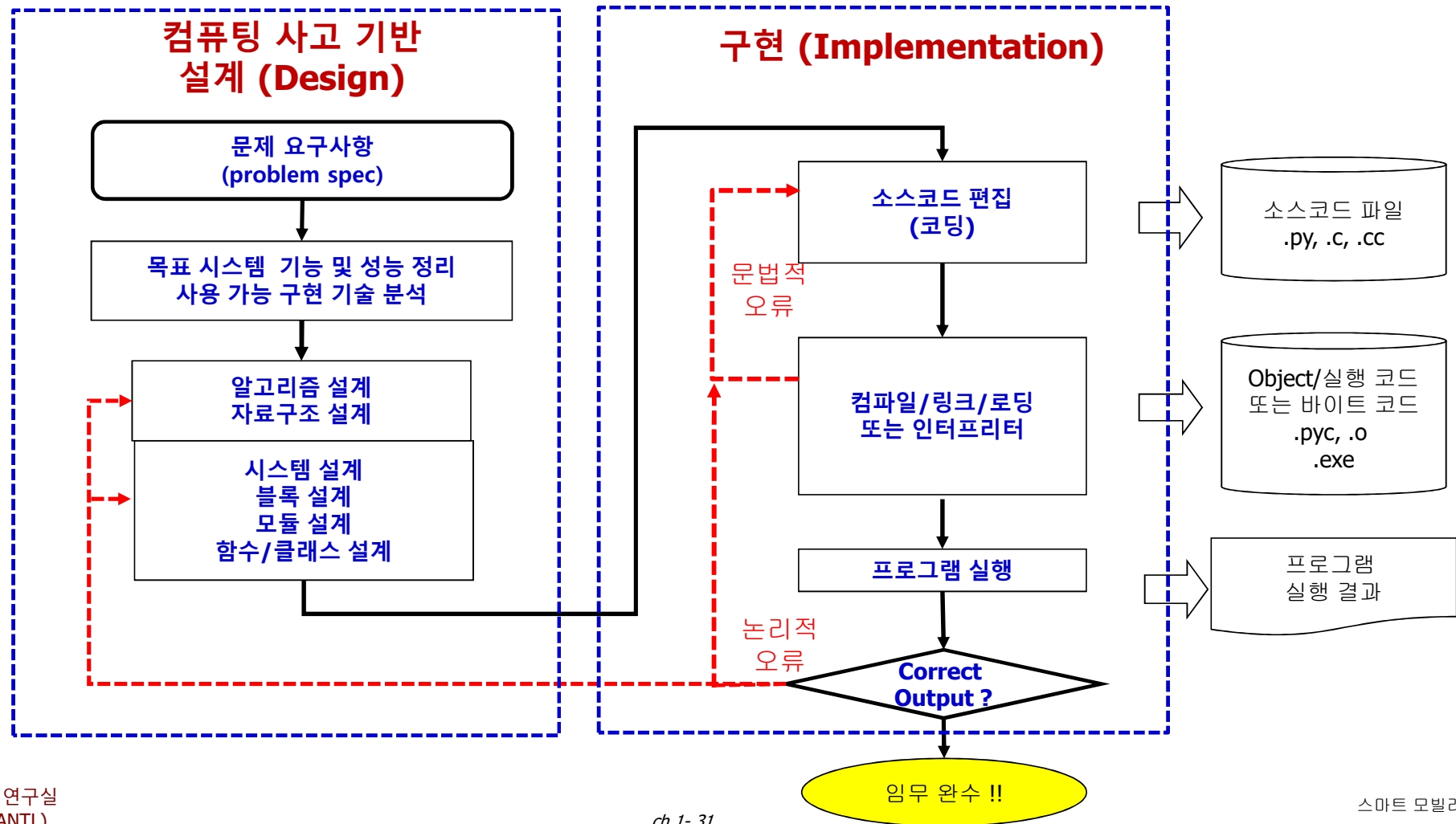
- 문제해결 방법에서 컴퓨터 장치와 프로그래밍을 활용하여 문제를 정확하고, 신속하며, 효율적으로 해결할 수 있도록 생각/사고
- 컴퓨터나 사람이 문제해결을 효과적으로 수행할 수 있도록 주어진 문제를 일반화 (generalize)/추상화 (abstract)하여 정의하고, 그 문제를 해결하는 방법을 체계적으로 정리하며, 이 문제와 유사한 다양한 변종 문제 해결에서도 사용할 수 있게 하는 사고 과정 (thinking process) 전체를 의미

컴퓨팅 사고

◆ 컴퓨터 기반의 문제해결 단계

단계	세부 실행 내용
문제 요구사항 분석	<ul style="list-style-type: none"> 주어진 문제의 요구사항을 분석(analyze)하고 논리적으로 타당한 요구인지 파악
주어진 큰 문제를 작은 문제로 분해	<ul style="list-style-type: none"> 주어진 큰 문제를 작은 문제/작업 단위로 분해 (decompose)
데이터 수집 분석, 규칙과 패턴 파악	<ul style="list-style-type: none"> 데이터를 수집하여 분석하고, 문제에 포함된 규칙과 패턴을 파악
모형화(modeling)/추상화(abstraction)	<ul style="list-style-type: none"> 규칙과 패턴을 기반으로 모형화/추상화 특징 추출과 일반화 (generalization)
알고리즘 구상 및 설계	<ul style="list-style-type: none"> 작은 단위 문제들에 대하여 이미 개발되어 있는 컴퓨터 기반 해결 방법을 활용하여 문제를 해결할 수 있도록 문제해결의 단계적 과정을 알고리즘으로 구성하고, 순서도나 유사코드로 표현
프로그램으로 구현 (코딩)	<ul style="list-style-type: none"> 구성된 알고리즘을 프로그램으로 구현하고 컴퓨터를 사용하여 모의시험 수행
실행 결과 확인, 분석	<ul style="list-style-type: none"> 모의시험 수행 결과를 측정 및 분석하여 당초 주어진 문제를 해결하는 방안이 되었는지 확인
알고리즘의 개선	<ul style="list-style-type: none"> 모의시험 수행 결과 분석에 따라 알고리즘을 개선 알고리즘 구상 및 설계를 보완하며, 구현 및 실험을 다시 실행

컴퓨팅 사고를 기반으로 한 프로그램 설계 및 구현 절차



현재 많이 사용되고 있는 프로그래밍 언어

◆ TIOBE Index for Jan 2023 (<https://www.tiobe.com/tiobe-index/>)

Jan 2023	Jan 2022	Change	Programming Language		Ratings	Change
1	1			Python	16.36%	+2.78%
2	2			C	16.26%	+3.82%
3	4	▲		C++	12.91%	+4.62%
4	3	▼		Java	12.21%	+1.55%
5	5			C#	5.73%	+0.05%
6	6			Visual Basic	4.64%	-0.10%
7	7			JavaScript	2.87%	+0.78%
8	9	▲		SQL	2.50%	+0.70%
9	8	▼		Assembly language	1.60%	-0.25%
10	11	▲		PHP	1.39%	-0.00%
11	10	▼		Swift	1.20%	-0.21%
12	13	▲		Go	1.14%	+0.10%
13	12	▼		R	1.04%	-0.21%
14	15	▲		Classic Visual Basic	0.98%	+0.01%
15	16	▲		MATLAB	0.91%	-0.05%
16	18	▲		Ruby	0.80%	-0.08%

컴퓨팅 사고와 프로그래밍 교육

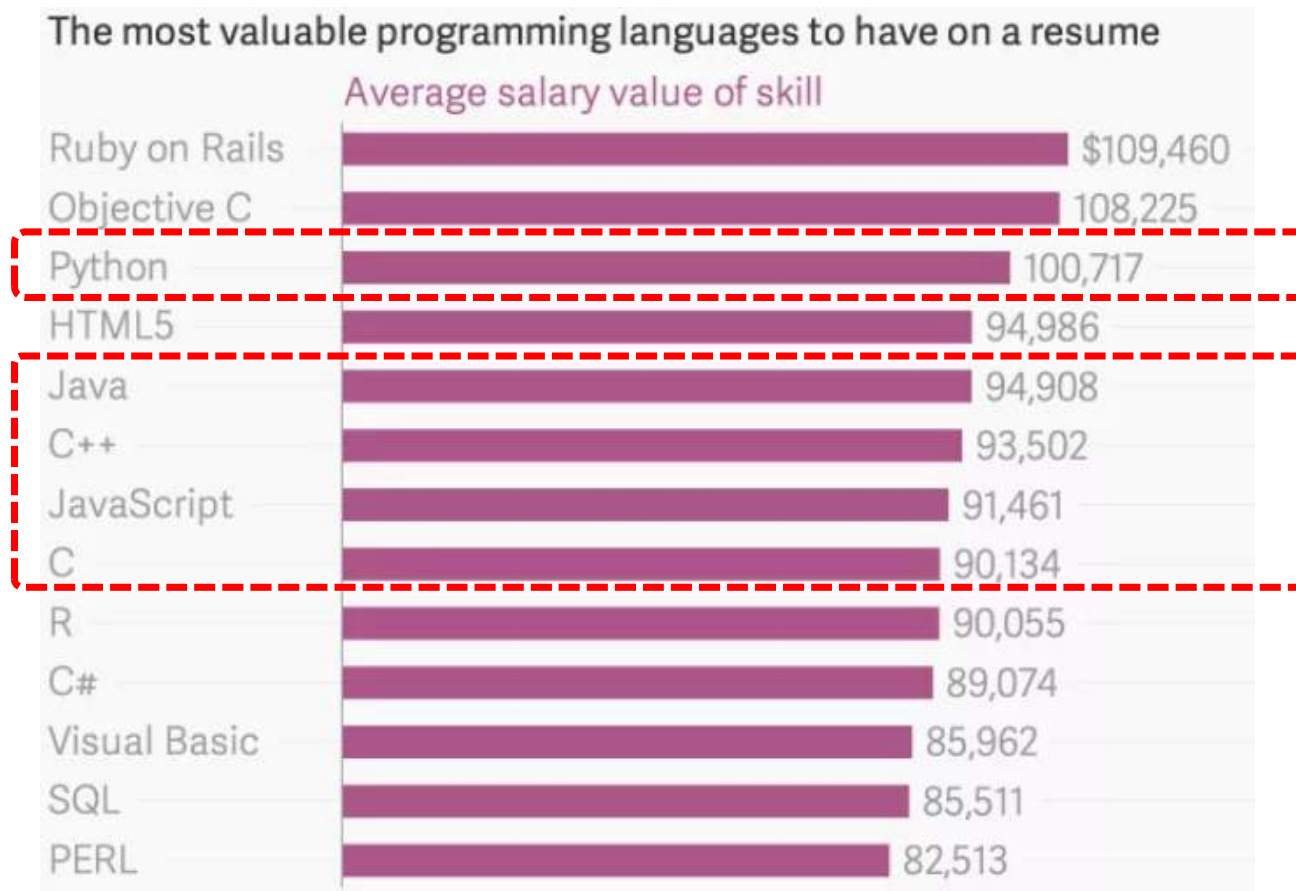
◆ 주요 프로그래밍 언어와 특성

프로그래밍 언어	특성
C	<ul style="list-style-type: none"> 하드웨어를 직접 제어하는 성능이 좋아 많은 운영체제 (operating system, OS)의 구현에 사용됨 절차 위주의 프로그래밍 구성, 컴파일러 사용
C++	<ul style="list-style-type: none"> 객체 지향형 (object-oriented)의 클래스와 객체 (object)를 사용하며, C의 기능도 함께 포함 컴파일러 사용
C#	<ul style="list-style-type: none"> .NET 환경에 최적화되어 있으며 컴포넌트 기반의 소프트웨어 개발 패러다임을 지원하는 컴포넌트 지향 프로그래밍 언어 단순한 콘솔 응용 프로그램부터 인터넷 및 분산환경 시스템, 모바일 시스템을 위한 응용 프로그램, 게임과 같은 멀티미디어 콘텐츠에 이르기까지 다양한 소프트웨어를 쉽게 제작할 수 있도록 설계된 범용 프로그래밍 언어 C++의 객체지향성과 자바의 분산환경 처리에 적합한 다중성을 지님
Java	<ul style="list-style-type: none"> 하드웨어 플랫폼에 상관없이 사용할 수 있다는 장점이 있음 객체지향형의 클래스와 객체 가능 사용 인터프리터 또는 컴파일러 사용 안드로이드 앱이 Java 기반이며, 웹 기반의 업무 시스템에서 Java가 많이 사용됨
Java Script	<ul style="list-style-type: none"> Java 스크립트 프로그래밍
Python	<ul style="list-style-type: none"> 객체 지향형 클래스와 객체 사용가능 인터프리터 사용
Visual Basic	<ul style="list-style-type: none"> 영상 처리 관련 전문 프로그래밍
Go	<ul style="list-style-type: none"> 2009년 Google이 개발한 프로그래밍언어. Garbage collection 기능 제공, concurrent (동시처리/병행처리) 지원
Swift	<ul style="list-style-type: none"> 애플의 iOS와 macOS를 위한 프로그래밍 언어 기존의 애플 운영체제용 언어인 오브젝티브-C와 함께 공존하도록 개발. 오브젝티브-C와 마찬가지로 LLVM으로 빌드되고 같은 런타임을 공유. 클로저, 다중 리턴 타입, 네임스페이스, 제네릭스, 타입 유추 등 오브젝티브-C에는 없었으나 현대 프로그래밍 언어가 갖고 있는 기능을 많이 포함시켰으며 코드 내부에서 C나 오브젝티브-C 코드를 섞어서 프로그래밍하거나 스크립트 언어처럼 실시간으로 상호작용하며 프로그래밍 할 수 있음
Rust	<ul style="list-style-type: none"> C/C++에서 Null Pointer에 의한 메모리 사용에서의 불안정한 부분을 ownership과 life time 개념으로 개선 Concurrent 동시처리/병렬처리 기능 지원



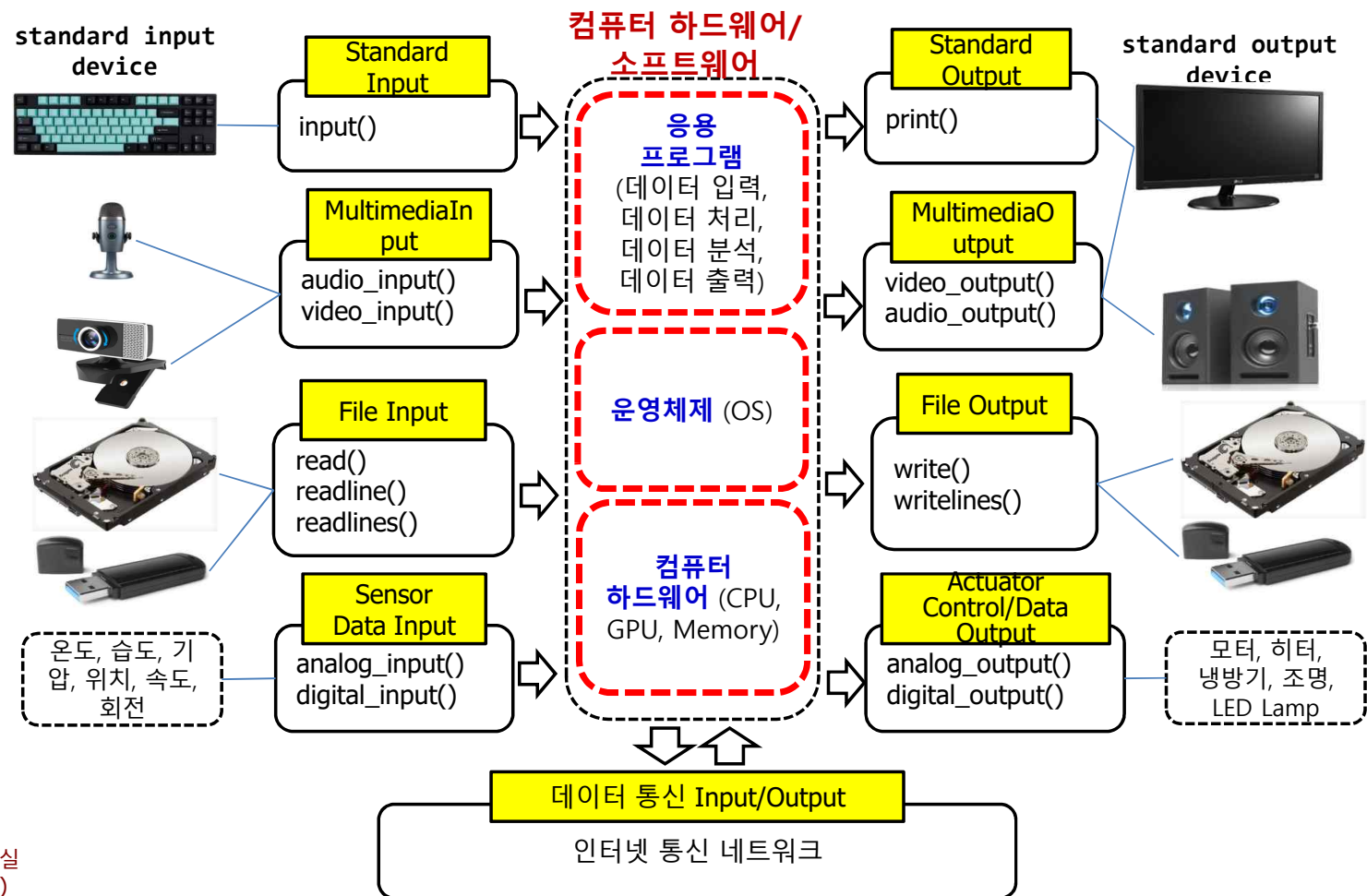
프로그래밍언어별 연봉

◆ <http://techneedle.com/archives/19283>

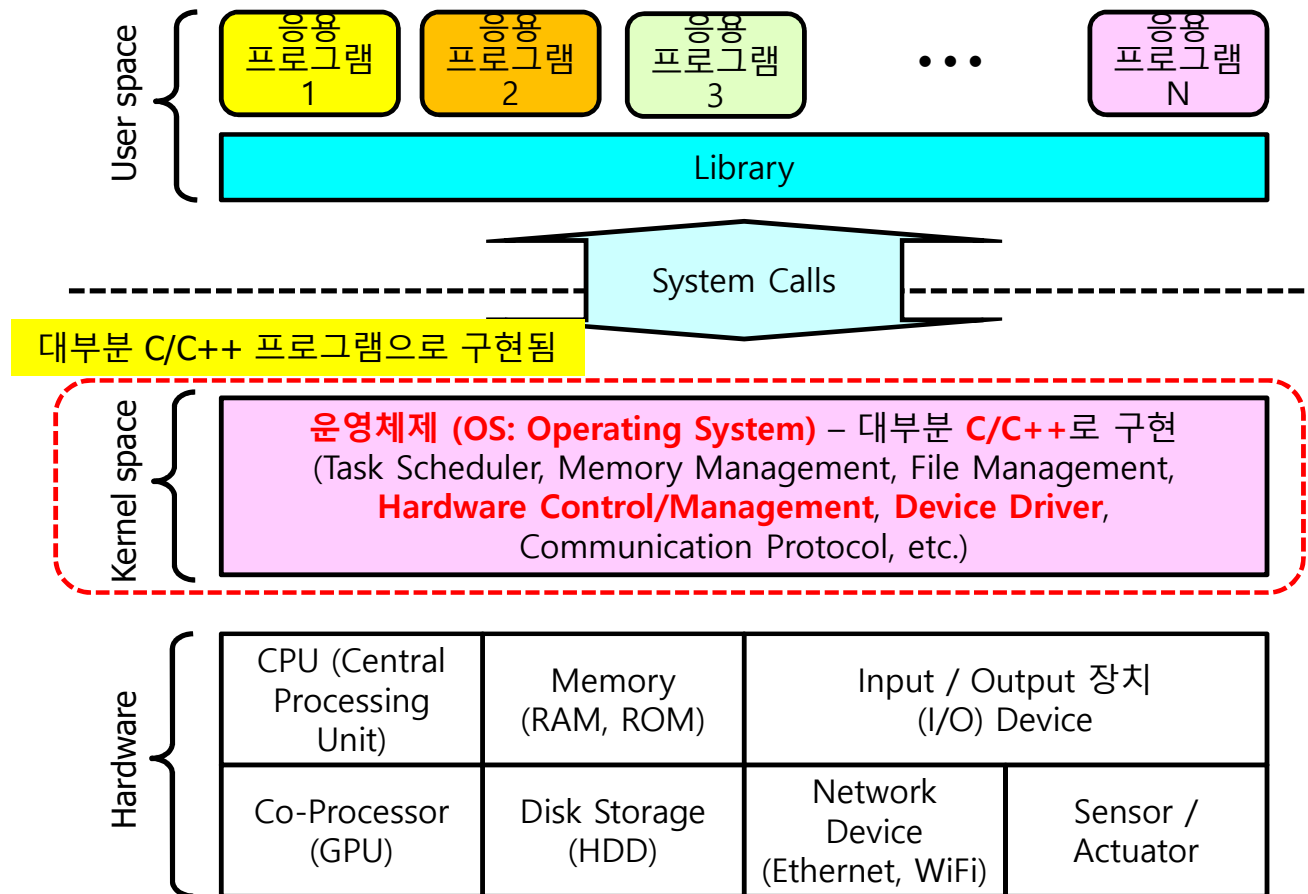


컴퓨터의 기본 기능

◆ 컴퓨터의 기본 기능



컴퓨터 기본 구조 – 하드웨어, 운영 체제 및 응용 소프트웨어



프로그램 실행과 데이터 처리

◆ 컴퓨터 프로그램의 실행 – 데이터 입력, 데이터 처리, 결과 출력 및 활용

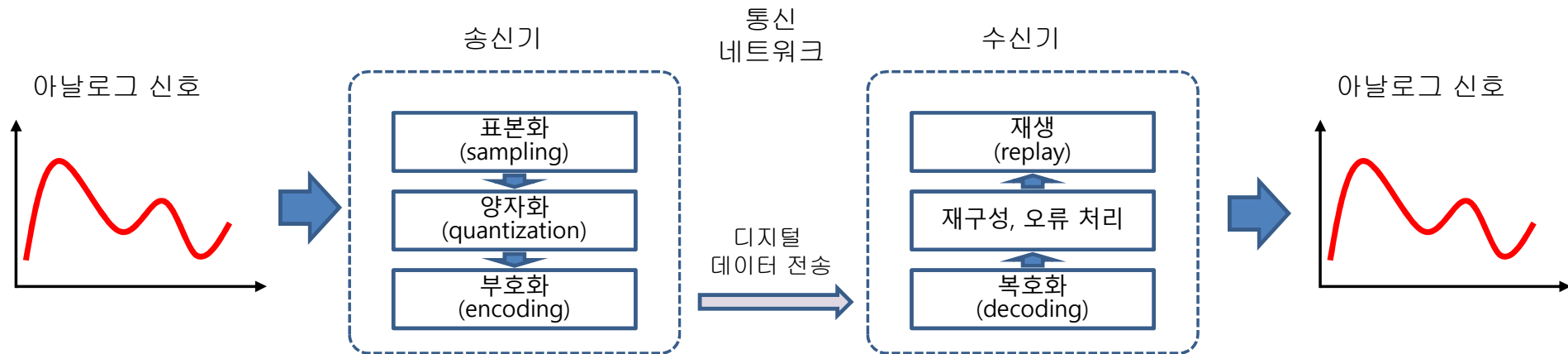
컴퓨터 프로그램 예	데이터 입력	데이터 처리	결과 출력 및 활용
스마트 온실의 온도/습도 제어	온도 센서 데이터 습도 센서 데이터	온도와 습도 데이터를 기준값과 비교, 통계 분석	액추에이터 (히터, 냉방기)의 구동
스마트 자율 주행	GPS 위치 데이터 차선 영상 데이터 도로 표지판 영상 데이터 근접 센서 데이터	차선 및 경로 유지 분석, 근접 차량 및 장애물 분석	속도 및 방향 제어
네비게이션의 최단 경로 탐색	목표 지점 GPS 위치 데이터	현재 위치에서 목표 지점까지의 최단 경로 탐색	목표지점까지의 경로 제어
스마트 단말 장치의 예측 문자열 제시	문자 입력	현재까지 입력된 문자열에 가능성이 높은 문자열들의 검색	현재까지 입력된 문자열에 가장 확률이 높은 상위 문자열 제시

컴퓨터에서의 정보 표현 – 아날로그와 디지털

아날로그와 디지털

◆ 아날로그와 디지털

구분	설명	예
아날로그 (analog)	<ul style="list-style-type: none"> 자연계의 대부분 신호 형태이며, 연속적인 값을 가짐 컴퓨터에서 아날로그 신호를 직접 처리하지는 않으며, 디지털 데이터로 변환하여 처리 	<ul style="list-style-type: none"> 사람 목소리 (100Hz ~ 3.4KHz) 온도, 습도, 기압
디지털 (digital)	<ul style="list-style-type: none"> bit (binary digit), byte/octet (8 bit) 단위 아날로그 데이터를 일정한 크기의 비트(bit) 수로 표현된 불연속적인 값을 가짐 사용된 비트 수에 따라 정확도가 달라지며, 원래 아날로그 값과 차이가 발생함 	<ul style="list-style-type: none"> 사람의 가청 주파수 영역 (20Hz~20KHz)의 무손실 음원에서는 24-비트/48~192KHz 샘플링으로 디지털 오디오 데이터 생성 숫자는 정수 (integer) (8~64비트), 실수(float, double) (32~128비트) 등으로 표현



컴퓨터에서 사용되는 데이터 표현 단위

◆ 비트 (bit)와 바이트 (byte)

데이터 표현 단위	비트 수	바이트 수	데이터 값 범위	설명
비트 (bit)	1		0 또는 1을 표시	On/Off 상태 표현
바이트 (byte)	8	1	0 ~ 255	최대 256가지의 값을 구분하여 표현

◆ 기본 숫자 자료형

기본 자료형	바이트 수	데이터 값 범위	주요 용도
문자 (character)	1	0 ~ 255	ASCII 문자 표현
	2	0 ~ 65535	unicode 문자 표현
정수 (integer)	1	0 ~ 255	부호 (sign)를 사용하지 않는 양수 범위의 정수
		-128 ~ +127	부호 (sign)를 사용하는 정수
	2	0 ~ 65535	양수 범위의 정수
		-32768 ~ +32767	부호 (sign)를 사용하는 정수
	4	0 ~ 4294967295	부호 (sign)를 사용하지 않는 양수 범위의 정수
		-2147483648 ~ +2147483647	부호 (sign)를 사용하는 정수
실수 (float)	4	$3.4 \times 10^{-38} \sim 3.4 \times 10^{38}$	부호 (sign): 1 비트 지수 (exponent): 8비트 유효자리: 23비트
	8	$1.79 \times 10^{-308} \sim 1.79 \times 10^{308}$	부호 (sign): 1 비트 지수 (exponent): 11비트 유효자리: 50비트



정수 (integer) 데이터의 표현

◆ 정수의 진법 단위 표현 - 10진수 (decimal), 2진수 (binary), 8진수 (octal)

10진수 (decimal)	2진수 (binary)	16진수 (hexadecimal)	8진수 (octal)
0	0000	0x0	00
1	0001	0x1	01
2	0010	0x2	02
3	0011	0x3	03
4	0100	0x4	04
5	0101	0x5	05
6	0110	0x6	06
7	0111	0x7	07
8	1000	0x8	10
9	1001	0x9	11
10	1010	0xA	12
11	1011	0xB	13
12	1100	0xC	14
13	1101	0xD	15
14	1110	0xE	16
15	1111	0xF	17

정수 (integer) 데이터의 표현

◆ 음수 (negative) 정수의 표현 – 2's complement (2의 보수)

10진수 값	2의 보수에 따른 8비트 단위 비트 값
127	0111 1111
126	0111 1110
2	0000 0010
1	0000 0001
0	0000 0000
-1	1111 1111
-2	1111 1110
-126	1000 0010
-127	1000 0001
-128	1000 0000

부호 (sign) 비트

0: 양수 (positive)

1: 음수 (negative)

컴퓨터에서 사용되는 문자 자료형 (1)

◆ 문자 (character) 자료형 - ASCII (American Standard Code for Information Interchange)

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0x10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x20	SP	!	“	#	\$	%	&	‘	()	*	+	,	-	.	/
0x30	0 (0x30)	1 (0x31)	2	3	4	5	6	7	8	9 (0x39)	: (0x3A)	;	<	=	>	?
0x40	@	A (0x41)	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x50	P	Q	R	S	T	U	V	W	X	Y	Z (0x5A)	[(0x5B)	\]	^	_
0x60	`	a (0x61)	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z (0x7A)	{ (0x7B)		}	~	DEL

컴퓨터에서 사용되는 문자 자료형 (2) - unicode

◆ Unicode

- <https://home.unicode.org/>
- 유니코드(영어: Unicode)는 전 세계의 모든 문자를 컴퓨터에서 일관되게 표현하고 다룰 수 있도록 설계된 산업 표준
- 유니코드의 목적은 현존하는 문자 인코딩 방법들을 모두 유니코드로 통합, 교체하려는 것임
- 유니코드에서 한글 코드 범위는 AC00 ~ D7AF 임 (<https://jjeong.tistory.com/696>)

U+AC00 to U+AD00

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UTF8: 234, 176, 128;	UNICODE: AC0	가	각	갈	갸	간	강	강	갇	갈	갸	갈	갈	갈	갈	갈	갈
UTF8: 234, 176, 144;	UNICODE: AC1	감	갈	갈	갸	갸	강	갸	갸	갈	갈	갈	개	개	개	개	개
UTF8: 234, 176, 160;	UNICODE: AC2	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 176, 176;	UNICODE: AC3	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 177, 128;	UNICODE: AC4	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈
UTF8: 234, 177, 144;	UNICODE: AC5	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈
UTF8: 234, 177, 160;	UNICODE: AC6	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈	갈
UTF8: 234, 177, 176;	UNICODE: AC7	거	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 178, 128;	UNICODE: AC8	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 178, 144;	UNICODE: AC9	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 178, 160;	UNICODE: ACA	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 178, 176;	UNICODE: ACB	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 179, 128;	UNICODE: ACC	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 179, 144;	UNICODE: ACD	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 179, 160;	UNICODE: ACE	고	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸
UTF8: 234, 179, 176;	UNICODE: ACF	곰	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸	갸

U+D700 to U+D800

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UTF8: 237, 156, 128;	UNICODE: D70	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 156, 144;	UNICODE: D71	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 156, 160;	UNICODE: D72	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 156, 176;	UNICODE: D73	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 157, 128;	UNICODE: D74	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 157, 144;	UNICODE: D75	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 157, 160;	UNICODE: D76	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 157, 176;	UNICODE: D77	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 158, 128;	UNICODE: D78	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 158, 144;	UNICODE: D79	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 158, 160;	UNICODE: D7A	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 158, 176;	UNICODE: D7B	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 159, 128;	UNICODE: D7C	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 159, 144;	UNICODE: D7D	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 159, 160;	UNICODE: D7E	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉
UTF8: 237, 159, 176;	UNICODE: D7F	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉	헉



파이썬 프로그래밍 언어

파이썬 프로그래밍 언어 개요

◆ 파이썬 (Python)

- 네덜란드 Guido van Rossum이 1990년대 초에 개발
- 영국 TV 프로그램 “Monty Python” 제목
- 공개 소프트웨어, general-purpose language
- 객체 지향형 개념을 사용하며, 일반적인 함수도 사용
- 인터프리터 사용 (컴파일러를 사용하여 사전에 실행코드를 만들지 않음)
- 동적 자료형 설정 (dynamically typed): 프로그램에서 사용되는 변수 (variable)의 자료형이 사전에 설정되지 않고, 실제 값이 대입될 때 결정됨
- 다양한 고급 자료형 및 자료구조를 지원: Lists, Tuples, Dictionaries (maps), Sets
- 스크립트 언어로서 간결함
- C/C++, Object-C, Java, Fortran 등의 언어로 개발된 모듈을 쉽게 통합시킬 수 있음



파이썬 프로그래밍 개요

◆ 파이썬 프로그래밍의 특징

- 프로그램의 구역을 들여쓰기 (indentation)으로 구분
- 편리한 시퀀스 자료형 제공
 - Strings '...': 문자열, 문자의 배열로 처리 가능
 - Lists [...]: 다양한 자료형의 원소들을 리스트로 구성
 - Tuples(...): 관련항목들을 (name, type, value)와 같은 형식으로 묶어서 사용하도록 함
- 시퀀스 자료형에 대하여 편리한 인덱싱과 슬라이싱 기능 제공
- 프로그램의 기능별로 함수를 구성하여 모듈화 시킬 수 있으며, 다양한 응용 프로그램에서 재사용 가능하게 함
- Java 프로그램 환경과 유사한 예외처리 (exception handling)
- C++ 및 Java의 객체 지향형 (object-oriented) 프로그래밍 지원
- 시퀀스 자료형에 대한 반복자 (iterator)를 제공하며, 코루틴 개념의 제네레이터 (generator) 함수 제공

파이썬 프로그래밍 개요

◆ 파이썬 프로그래밍의 장점?

- 스크립트 기반의 프로그래밍으로 쉽게 배울 수 있음
- “Pythonic” 스타일은 매우 간결한 프로그램 구성 가능
- 객체 지향형 프로그래밍의 장점을 활용하지만, C++나 Java보다 쉽게 사용할 수 있는 환경 제공
 - 파이썬에서 사용하는 모든 값과 개체들은 객체 (object)임
- 동적 자료형 (dynamic typing)을 사용하여 프로그램의 일반화/추상화의 장점을 활용
 - 시퀀스, 딕셔너리, 셀, 튜플 등의 고급 자료형을 다양한 기본 자료형에 활용할 수 있게 함
 - 파이썬 고급 자료형을 사용하여 복잡한 응용 자료구조 및 알고리즘을 일반화 (generalized)/추상화 (abstraction) 형식으로 구현 가능

파이썬 프로그래밍 개요

◆ 동적 자료형 설정 (Dynamic typing)

● 정적 자료형 설정 : C/C++, Java

- 프로그램에서 사용되는 변수(variable)들의 자료형은 소스코드 작성 단계에서 미리 설정 됨
- 변수들의 사용에서 정확한 자료형의 사용을 엄격하게 관리함
- 함수호출이나 변수를 사용한 대입 (assignment) 등에서 자료형이 서로 다른 경우 문법 오류 발생

● 동적 자료형: Python

- 변수 (variable)들은 처음 대입이 될 때 생성되며, 대입되는 데이터에 따라 자료형이 동적으로 결정됨
- 하나의 변수가 다양한 자료형의 객체로 대입될 수 있음
- 모든 자료형이 동일한 방식으로 관리됨
- **문제점 (main drawback)**: 자료형에 관련된 오류가 실행되는 단계에서만 확인될 수 있음

파이썬 자료형

◆ 파이썬 내장 자료형 (built-in data type)

자료형		변경 가능 여부 (mutable)	순차 반복 접근 가능 여부(iterable)	사용 예
불리언 (boolean)	bool	불가능	불가능	True, False
숫자 (numeric)	int	불가능	불가능	정수형
	float	불가능	불가능	실수형
	complex	불가능	불가능	복소수
시퀀스 (sequence)	str	불가능	가능	"abcdefg"
	bytes	불가능	가능	b'\0x00\0x01\0x02\0x03'
	bytearray	가능	가능	b'\0x00\0x01\0x02\0x03'
	memoryview	가능/불가능	가능	
	list	가능	가능	[0, 1, 2, 3],
	tuple	불가능	가능	(1, 2, 1, 2, 3)
	range	불가능	가능	range(start, end[, step])
매핑 (mapping)	dict	가능	가능	{1:'A', 2:'B', 3:'C'}
집합 (set)	set	가능	가능	{1, 2, 3, 4, 5}
	frozenset	불가능	가능	{1, 2, 3, 4, 5}

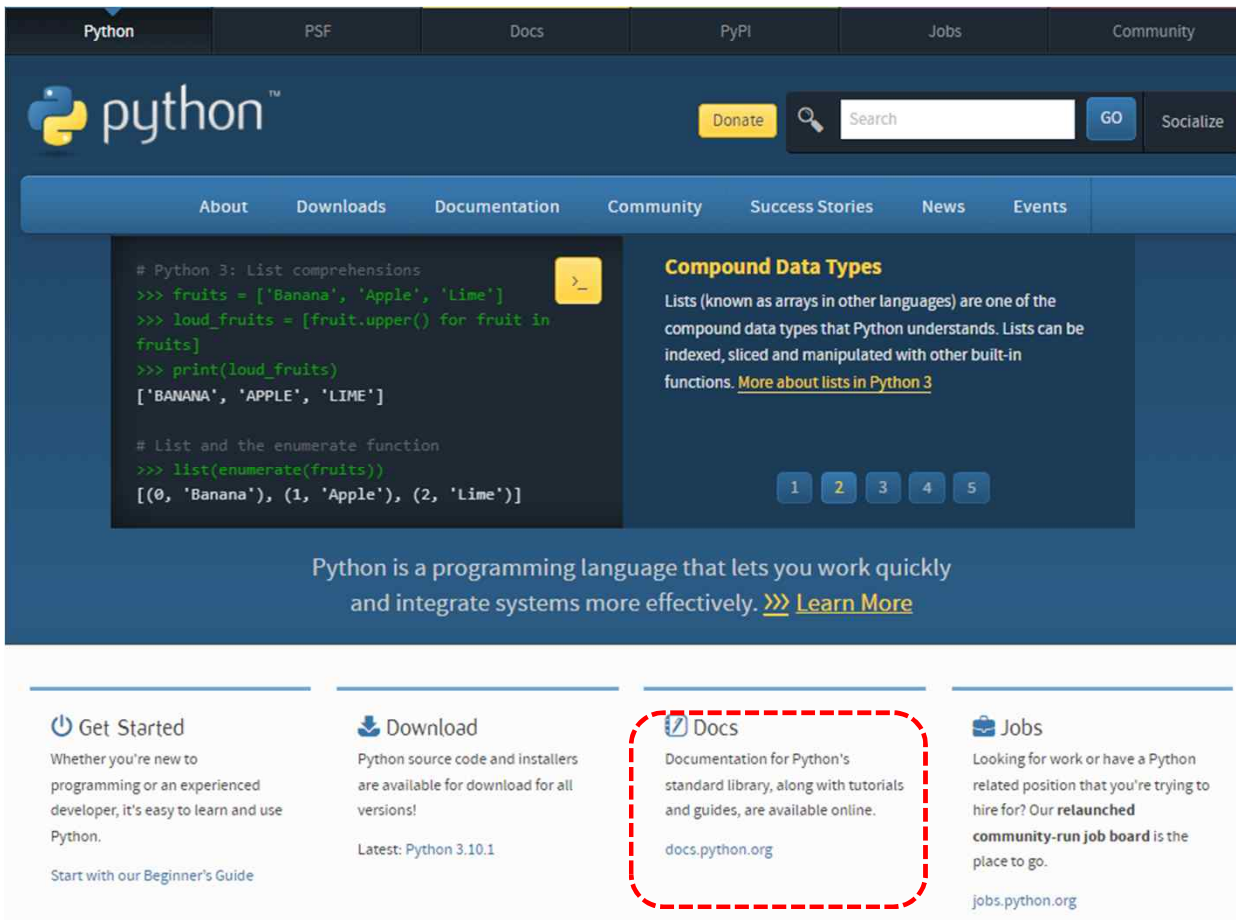
파이썬 프로그래밍 환경

◆ 파이썬 프로그래밍 환경 설치

- Downloads: <http://www.python.org>
- Documentation: <http://www.python.org/doc/>
- Free book: <http://www.diveintopython.org>



www.python.org



The screenshot shows the Python.org homepage. At the top, there's a navigation bar with links: Python, PSF, Docs, PyPI, Jobs, and Community. Below this is the Python logo and a search bar. A secondary navigation bar includes links: About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a code snippet on the left and a text block on the right titled "Compound Data Types". The code snippet shows list comprehensions and the enumerate function. The text block explains that lists are compound data types and provides a link to "More about lists in Python 3". Below this, there are five numbered buttons (1-5). At the bottom, there's a footer with four sections: "Get Started", "Download", "Docs" (highlighted with a red dashed border), and "Jobs".

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Compound Data Types

Lists (known as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

1 2 3 4 5


Python is a programming language that lets you work quickly and integrate systems more effectively. [>>> Learn More](#)

Get Started
Whether you're new to programming or an experienced developer, it's easy to learn and use Python.
[Start with our Beginner's Guide](#)

Download
Python source code and installers are available for download for all versions!
Latest: Python 3.10.1

Docs
Documentation for Python's standard library, along with tutorials and guides, are available online.
docs.python.org

Jobs
Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job board** is the place to go.
jobs.python.org



The screenshot shows the Python 3.10.1 release announcement page. It features the Python logo and the text "Python 3.10.1" with the release date "Dec. 6, 2021". Below this is a large, stylized graphic of the number "3.10" with a snake-like head forming the "0". The graphic is surrounded by text: "BETTER TYPING SYNTAX", "SHIPPED", "BETTER", "STRUCTURAL PATTERN MATCHING", "BETTER ERROR MESSAGES", and "PARENTHESED CONTEXT MANAGERS".

Python 3.10.1

Release Date: Dec. 6, 2021

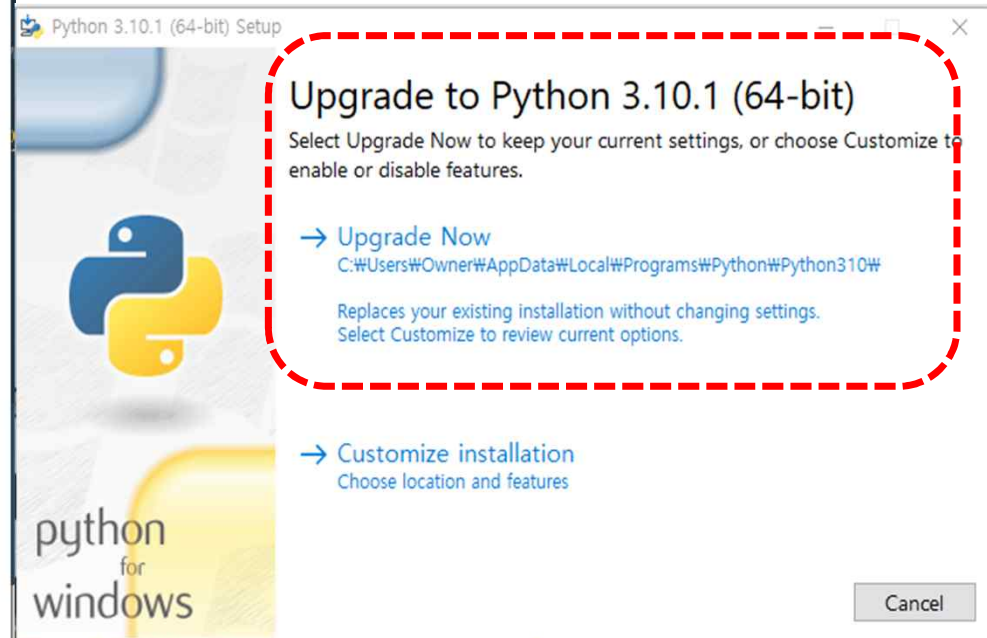
BETTER TYPING SYNTAX
SHIPPED
BETTER
STRUCTURAL PATTERN MATCHING
BETTER ERROR MESSAGES
PARENTHESED CONTEXT MANAGERS



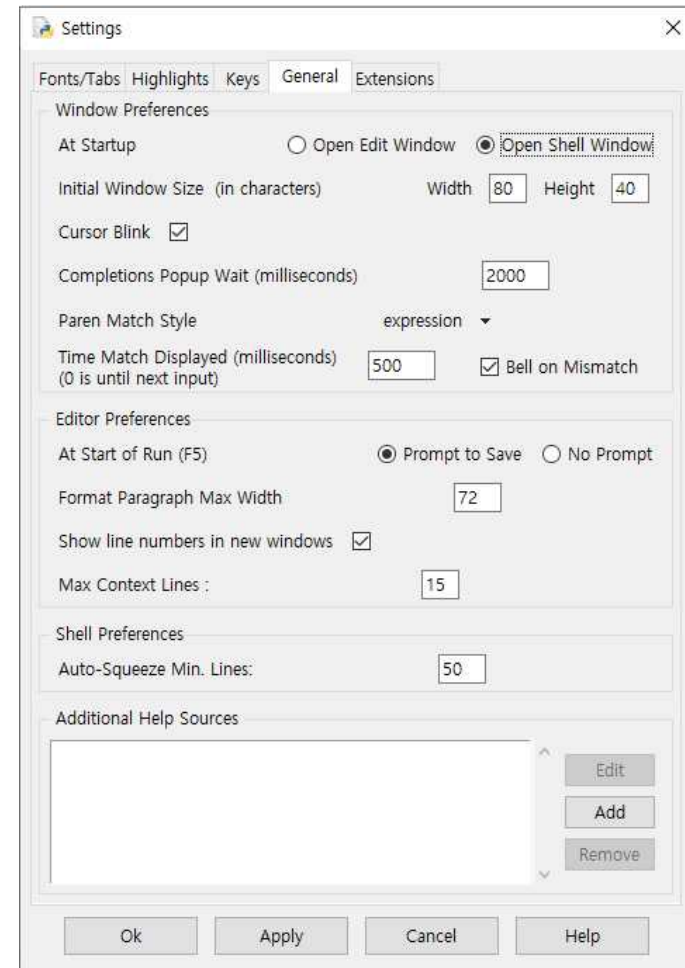
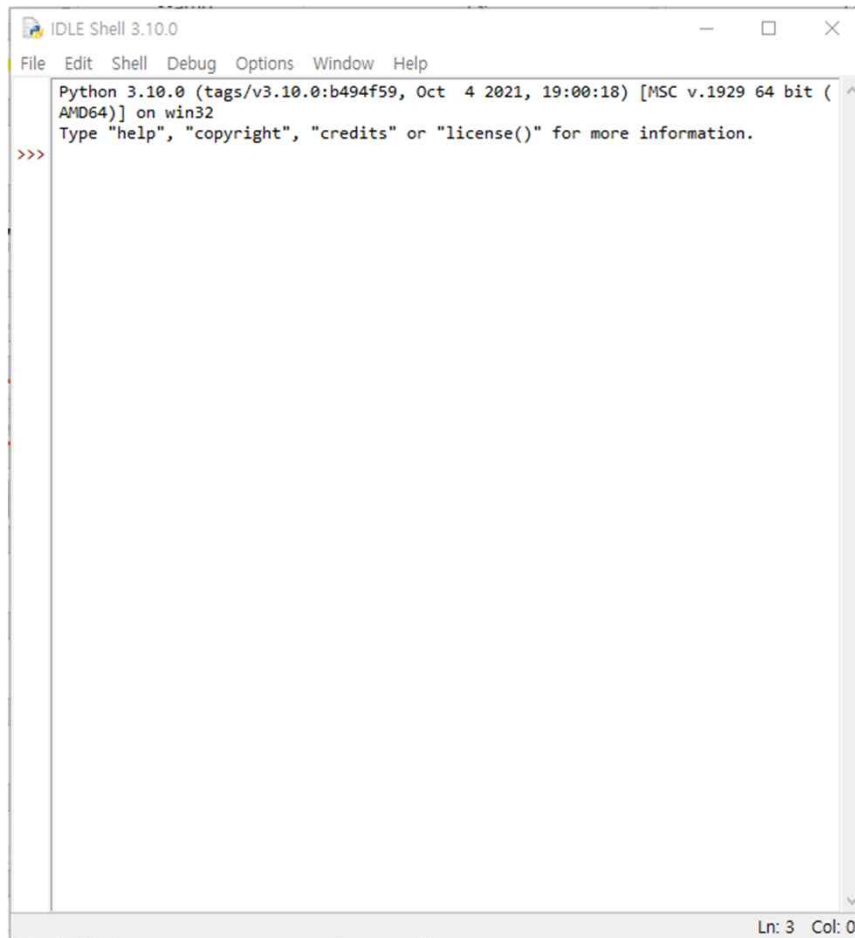
파이썬 최신 버전 설치

◆ Python 3.10 버전 설치

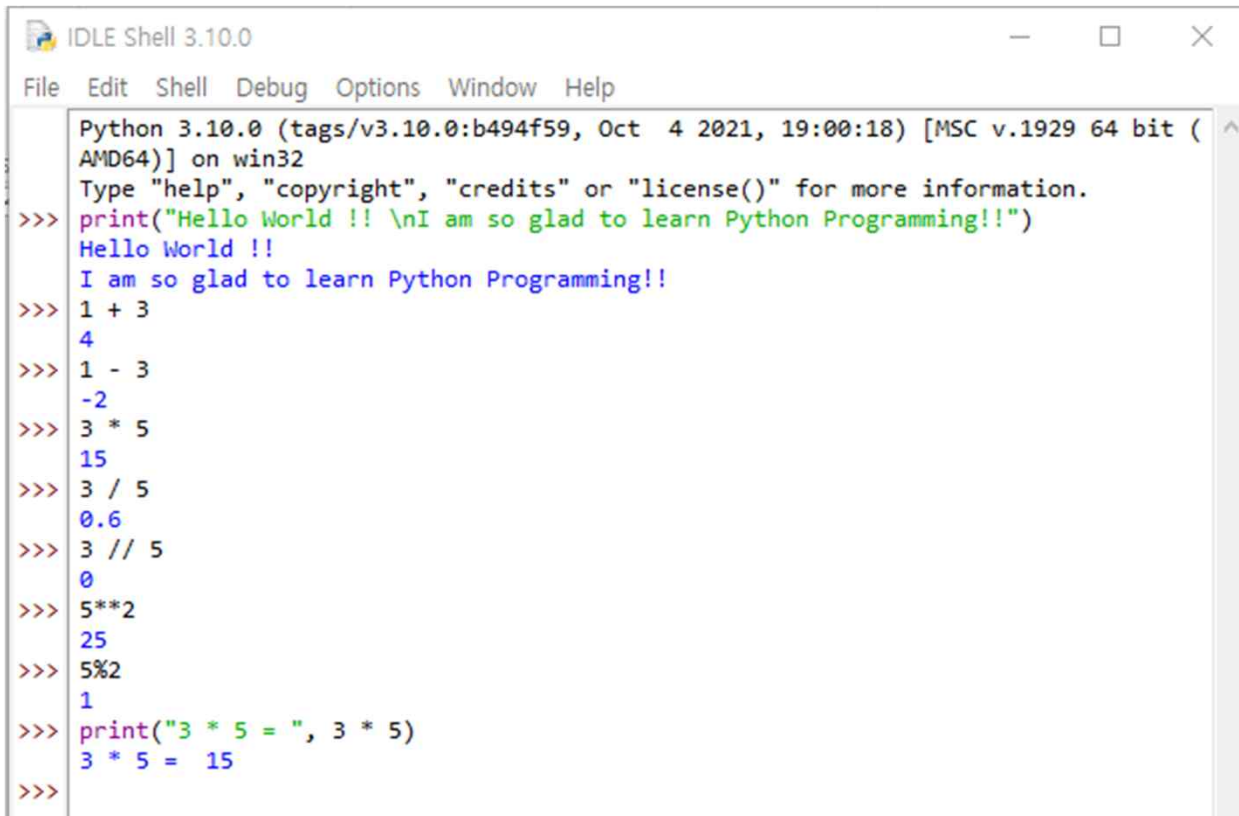
- 참고 : 3.11버전은 인공지능/기계학습 확장 패키지 설치에 문제 있을 수 있음



파이썬 셸 (Shell)



파이썬 셸 (Shell)에서의 간단한 프로그램 실행

A screenshot of the IDLE Shell 3.10.0 window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the following text:

```
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello World !! \nI am so glad to learn Python Programming!!")
Hello World !!
I am so glad to learn Python Programming!!
>>> 1 + 3
4
>>> 1 - 3
-2
>>> 3 * 5
15
>>> 3 / 5
0.6
>>> 3 // 5
0
>>> 5**2
25
>>> 5%2
1
>>> print("3 * 5 = ", 3 * 5)
3 * 5 = 15
>>>
```



파이썬 프로그래밍 환경 (1)

◆ 파이썬 셸 (shell)

```
*IDLE Shell 3.10.0*
File Edit Shell Debug Options Window Help
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> help()

Welcome to Python 3.10's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the internet at https://docs.python.org/3.10/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules. To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics". Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> keywords

Here is a list of the Python keywords. Enter any keyword to get more help.

False      class      from       or
None       continue  global     pass
True       def       if         raise
and        del       import     return
as         elif      in         try
assert    else      is         while
async     except   lambda    with
await     finally  nonlocal  yield
break     for      not
```

```
help> symbols

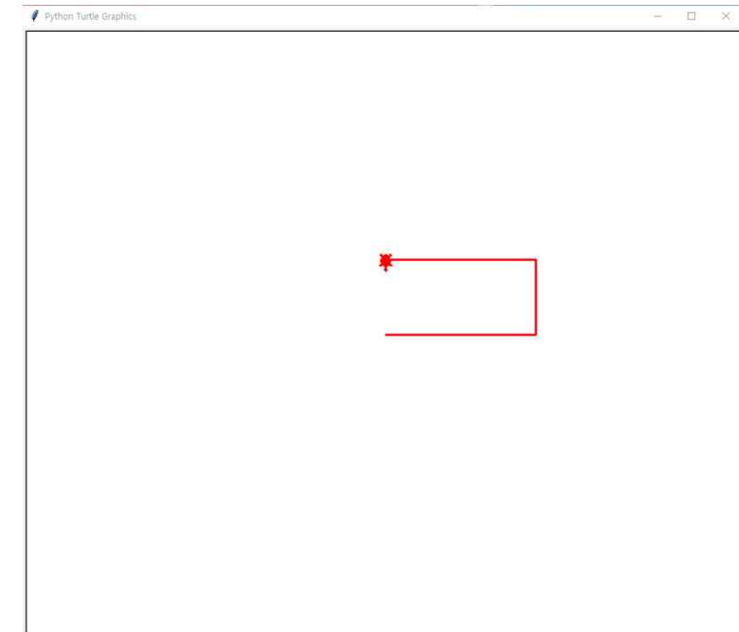
Here is a list of the punctuation symbols which Python assigns special meaning
to. Enter any symbol to get more help.

!=          +          <=         ,\
"           +=         <>         b"
""          ,          ==         b'
%           -          >          f"
%=          -=         >=         f'
&          .          >>         j
&=         ...        >>=        j
'           /          @          r"
...         //         J          r'
(           //=        [          u"
)           /=         \          u'
*           :          ]          |
**          <          ^          |=
**=         <<         ^=         ~
*=          <<=        -
```

파이썬 쉘 환경에서의 프로그래밍

◆ turtle graphic

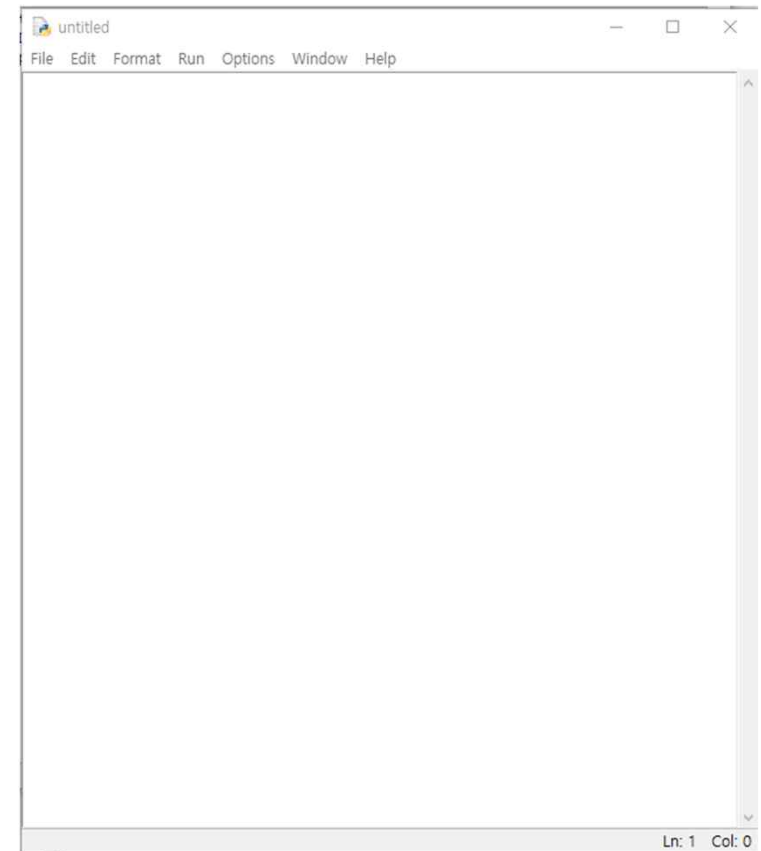
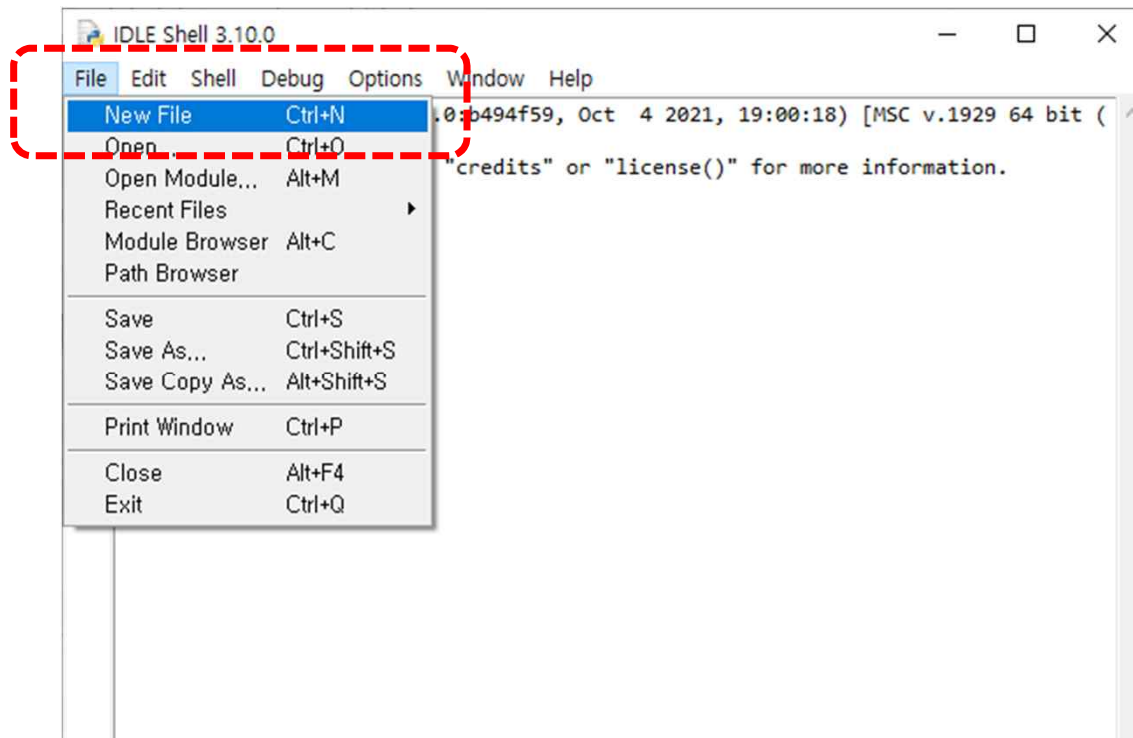
```
IDLE Shell 3.10.0
File Edit Shell Debug Options Window Help
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import turtle
>>> t = turtle.Turtle()
>>> t.shape("turtle")
>>> t.color("red")
>>> t.width(3)
>>> t.forward(200)
>>> t.left(90)
>>> t.forward(100)
>>> t.right(90)
>>> t.backward(200)
>>> t.right(90)
>>>
```



Python Shell

◆ File Menu

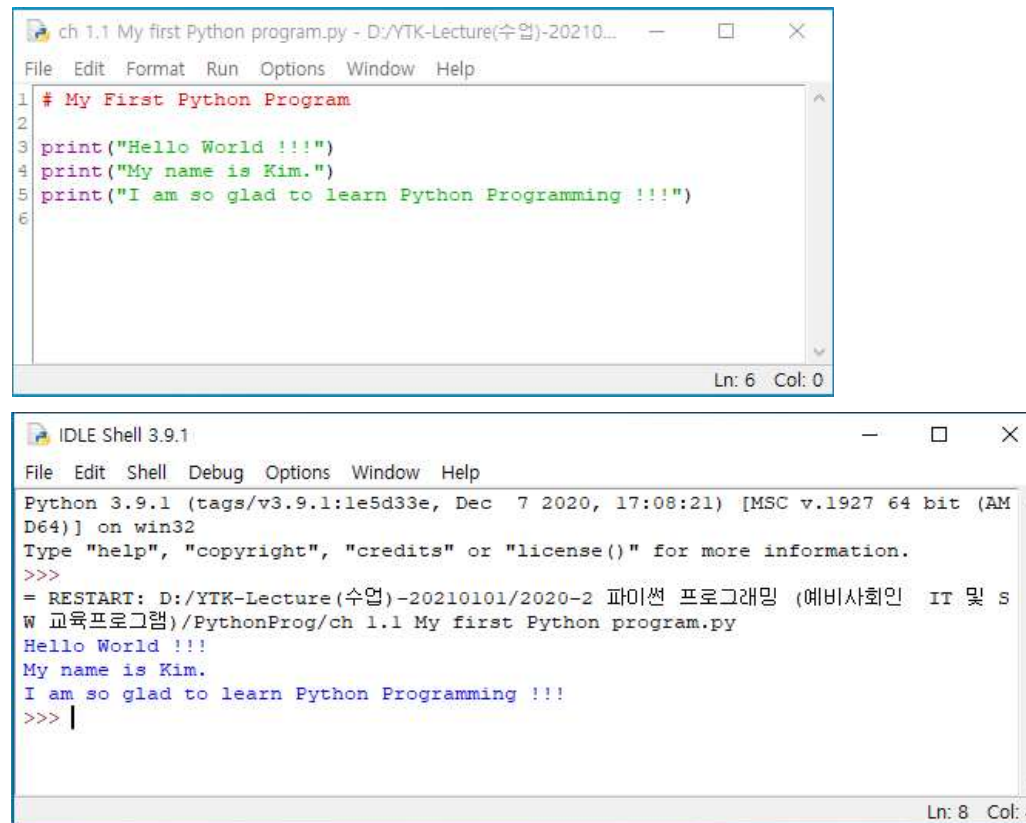
● New File



IDLE (Integrated Development and Learning Environment) 에서 파이썬 프로그램 작성 및 실행

◆ My First Python Program

- Save as (Ctrl+shift+S)
- Save (Ctrl+S)
- Run module (F5)



The screenshot displays two windows from the IDLE Python IDE. The top window, titled 'ch 1.1 My first Python program.py', contains a Python script with the following code:

```
1 # My First Python Program
2
3 print("Hello World !!!")
4 print("My name is Kim.")
5 print("I am so glad to learn Python Programming !!!")
6
```

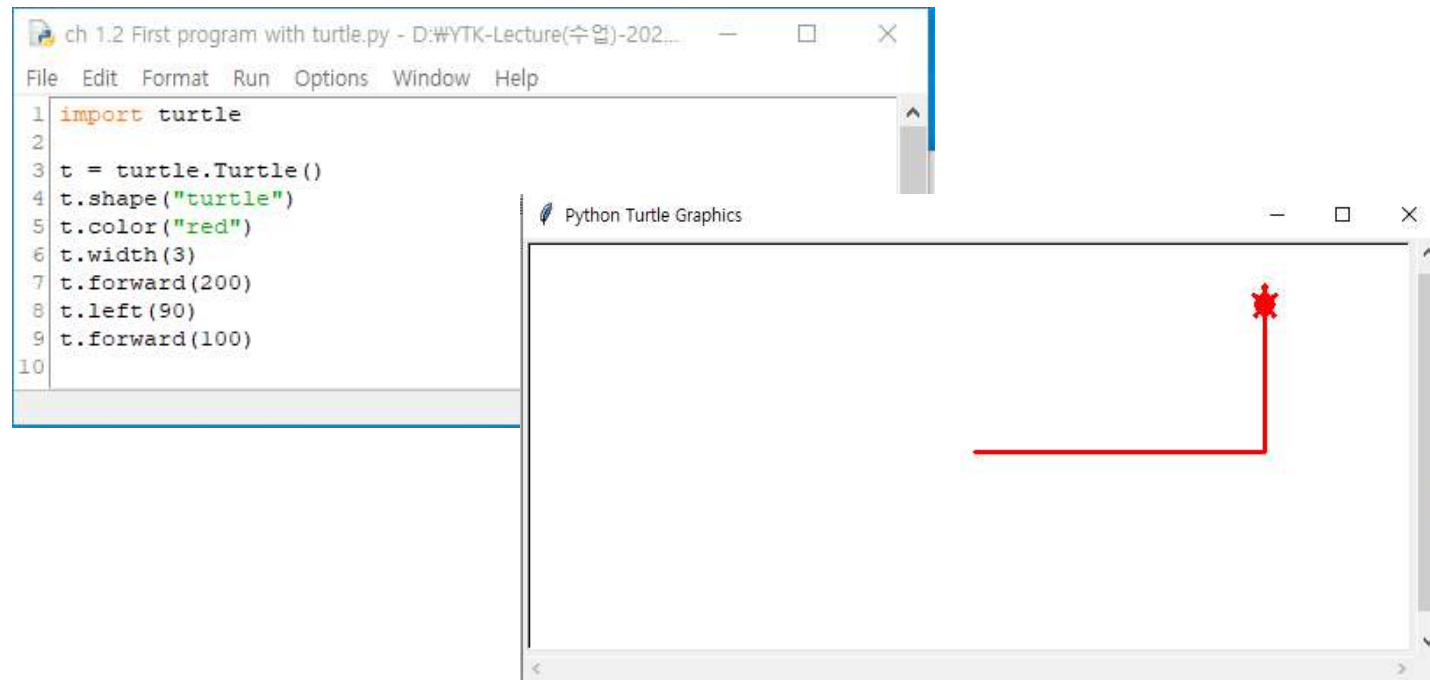
The bottom window, titled 'IDLE Shell 3.9.1', shows the output of running the script. It includes the Python version information and the execution of the print statements:

```
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:/YTK-Lecture(수업)-20210101/2020-2 파이썬 프로그래밍 (예비사회인 IT 및 s
W 교육프로그램)/PythonProg/ch 1.1 My first Python program.py
Hello World !!!
My name is Kim.
I am so glad to learn Python Programming !!!
>>> |
```

IDLE에서의 Turtle Graphic 예제

◆ My First Python Turtle Graphic Program

- Save as (Ctrl+shift+S)
- Save (Ctrl+S)
- Run module (F5)



파이썬 IDLE 환경에서의 프로그램 구현 및 실행

◆ Simple script editing, save and execute

- IDLE [File] → [New File]
- [File] → [Save]
- [Run] → [Run Module]
- [File] → [Open]

◆ 단축키

- 저장 (save) : Ctrl + S
- 이름지정 저장 (save as) : Ctrl+Shift+S
- 실행 (run module) : F5

파이썬 프로그램의 기본 구성

MyFirstPythonProgram.py

```
# MyFirstPythonProgram.py (1)
"""
Project : My first Python Program
Author: Cul-Soo Kim
Date of last update: Feb. 6, 2023
Update list:
    - v1.0 : Feb. 6, 2023
        . My first Python program with simple
          prints of variables
    - v1.1 : Feb. 6, 2023
        . Include str and mathematical operations
"""

print("This is my first Python program !")
my_name = "Chul-Soo Kim" # string variable
print("My name is", my_name, ".")
print("I am really happy to learn\
Python Programming !!")

# definitions of variables
x = 5 # integer variable
y = 2 # integer variable
print("x : ", x)
print("y : ", y)
```

```
# MyFirstPythonProgram.py (2)

# operations
sum_xy = x + y
sub_xy = x - y
mul_xy = x * y
div_xy = x / y
int_div_xy = x // y
print("x + y : ", sum_xy)
print("x - y : ", sub_xy)
print("x * y : ", mul_xy)
print("x / y : ", div_xy)
print("x // y : ", int_div_xy)
```

```
This is my first Python program !
My name is Chul-Soo Kim .
I am really happy to learn Python Programming !!
x : 5
y : 2
x + y : 7
x - y : 3
x * y : 10
x / y : 2.5
x // y : 2
```



기본 주석문 (basic comments)

◆ 주석문 (Comments)

- 프로그램의 목적 및 주요 기능
- 프로그램 작성자 및 작성 일자, 보완 일자
- 프로그램의 주요 수정/보완 내용

파이썬 프로그램 주석문 형식	설 명
<pre>""" comment_line_1 comment_line_2 """</pre>	여러 줄에 걸친 주석문
<pre># comment to the end of line</pre>	'#' 문자로부터 그 줄 끝까지 주석문

```
"""
Project : My first Python Program
Author: Cul-Soo Kim
Date of last update: Feb. 6, 2023
Update list:
- v1.0 : Feb. 6, 2023
    . My first Python program with simple
      prints of variables
- v1.1 : Feb. 6, 2023
    . Include str and mathematical operations
"""

# First Greetings
x = 5  # variable
```



데이터 입력 기능을 가지는 파이썬 프로그램 예제

◆ 데이터 입력 및 연산 예제

```
"""
    Basic Comments (same as before)
    """

# definitions of variables
x_str = input("input x = ") # variable
y_str = input("input y = ") # variable
x, y = int(x_str), int(y_str)
print("x, y = {}, {}".format(x, y))

# operations
sum_xy = x + y
sub_xy = x - y
mul_xy = x * y
div_xy = x / y
int_div_xy = x // y
print("x + y : ", sum_xy)
print("x - y : ", sub_xy)
print("x * y : ", mul_xy)
print("x / y : ", div_xy)
print("x // y : ", int_div_xy)
```

```
input x = 7
input y = 3
x, y = 7, 3
x + y : 10
x - y : 4
x * y : 21
x / y : 2.3333333333333335
x // y : 2
```

변수 (variable)의 선언 및 사용

◆ 변수 (variable) 선언 및 사용 예

```
# Python program variables
```

```
x = 1 # integer
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = 2.345 #float
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = [1, 2, 3, 4] # list
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = (7, 8, 9, 10) # tuple
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = {'A':1, 'B':2, 'C':3} # dict
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = {1, 2, 3, 4} # set
print("x = ", x)
print("type(x) = ", type(x))
```

```
--
x = 1
type(x) = <class 'int'>
x = 2.345
type(x) = <class 'float'>
x = [1, 2, 3, 4]
type(x) = <class 'list'>
x = (7, 8, 9, 10)
type(x) = <class 'tuple'>
x = {'A': 1, 'B': 2, 'C': 3}
type(x) = <class 'dict'>
x = {1, 2, 3, 4}
type(x) = <class 'set'>
```



상수 (constant)의 선언 및 사용

◆ 상수 (constant)의 선언 및 사용 예

```
# variables and constants in Python program

PI = 3.141592

radius = 5.0
circle_area = radius * radius * PI
print("Area of circle (radius : {}) : {}".format(radius, circle_area))

width = 5.0
length = 4.0
rectangle_area = width * length
print("Area of rectangle (width: {}, length: {}) : {}".format(width, length, rectangle_area))

base = 5.0
height = 4.0
triangle_area = (base * height) / 2.0
print("Area of triangle (base: {}, height: {}) : {}".format(base, height, triangle_area))

Area of circle (radius : 5.0) : 78.5398
Area of rectangle (width: 5.0, length: 4.0) : 20.0
Area of triangle (base: 5.0, height: 4.0) : 10.0
```



과학 기술 계산에서 많이 사용되는 상수들

◆ 기호상수 (symbolic constant) 정의

기호 상수의 예	설 명
$\text{PI} = 3.141592653589793238$	원주율의 값을 의미하며, 원면적, 원둘레, 삼각함수 등에서 사용
$\text{INT_MAX} = 2147483647$	32비트로 표현되며 부호가 포함된 정수 중 가장 큰 값
$\text{INT_MIN} = -2147483638$	32비트로 표현되며 부호가 포함된 정수 중 가장 작은 값
$\text{RAND_MAX} = 32767$	16비트값으로 생성되는 난수 (random number)의 최대 값

파이썬 프로그램의 식별자 (Identifier),
기본 수학연산, 함수호출

파이썬 프로그램 식별자

◆ 식별자 (identifier)

- 변수 (variable), 상수 (constant), 함수 (function) 이름
- 의미를 전달할 수 있는 영어 단어 사용

◆ 식별자의 예

구분		식별자 예
기 하, 도 형	원	radius (반지름), diameter (지름), circumference(원둘레), height (원기둥 높이)
	삼각형	base(밑 변), side (대 각 선), vertex (꼭 지 점), height (삼 각 형 높 이), prism (삼 각 기 둥), prism_height(삼각기둥 높이)
	사각형	width(가로), length(세로), height(사각기둥 높이)
도형 그리기		draw(그리기), rotate(회전), move(이동)
산술 계산		add (덧셈), subtract(뺄셈), multiply(곱셈), divide(나눗셈), modulo(모듈로)

식별자 (Identifier) 설정 원칙

◆ 식별자 설정 규정 (Rule of identifier)

- 영문자(대문자, 소문자), 밑줄 문자, 숫자로 구성
 - 대문자 'A' ~ 'Z', 소문자 'a' ~ 'z', 밑줄 문자 ('_'), 숫자 ('0' ~ '9')
- 숫자는 식별자의 첫 문자가 될 수 없음 ('0' ~ '9')
- 파이썬의 키워드 (keyword)는 식별자가 될 수 없음
- 식별자의 길이에 제한이 없음
- 대문자와 소문자를 구분함

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', '
def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if',
'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'retu
rn', 'try', 'while', 'with', 'yield']
>>> |
```


식별자 설정 방법

◆ 식별자 설정 방법

- Camel casing: 단어의 첫 문자를 대문자로 구분
- GNU 작명 관례: 단어 사이에 밑줄 문자 사용

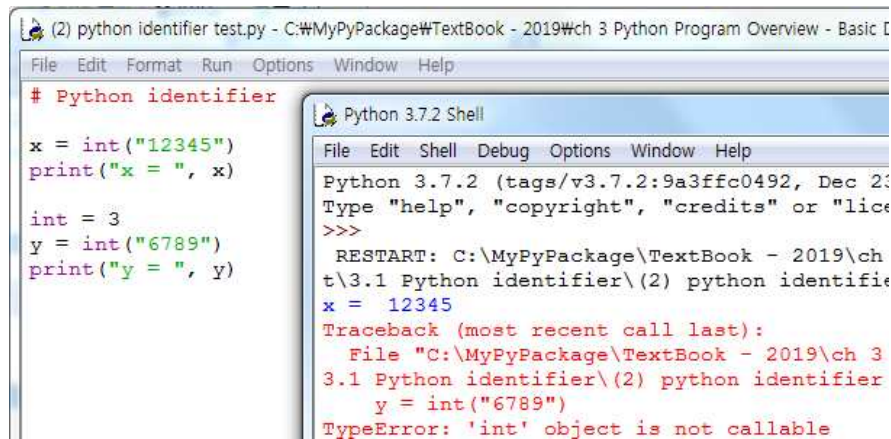
식별자 이름 작명 방법	설 명
camel casing	단어들을 연속적으로 나열하며, 각 단어의 첫 문자를 대문자로 표시 (단 첫 번째 단어는 소문자로 사용가능) 예) printComplex(), drawRectangle(), drawPolygon()
GNU 작명 관례	단어들을 연속적으로 나열하며, 각 단어사이에 밑줄 문자를 삽입 예) print_complex(), draw_rectangle(), draw_polygon()

식별자의 예

◆ 파이썬 식별자의 예

파이썬 식별자 예	사용 가능 여부	설명
average, avg, avg1	사용 가능	문자와 숫자
num_data, numData,	사용 가능	문자, 밑줄
ch, ch1, ch_1	사용 가능	문자, 밑줄
i, d, f	사용 가능	
_avg, __area	사용 가능	밑줄, 문자
NUM_DATA	사용 가능	대문자, 밑줄
auto, char, int, class, while, for	사용 불가능	keyword
3avg	사용 불가능	숫자로 시작
avg!, avg%, &avg, dollar\$	사용 불가능	특수문자 포함

잘못 설정된 파이썬 식별자의 예



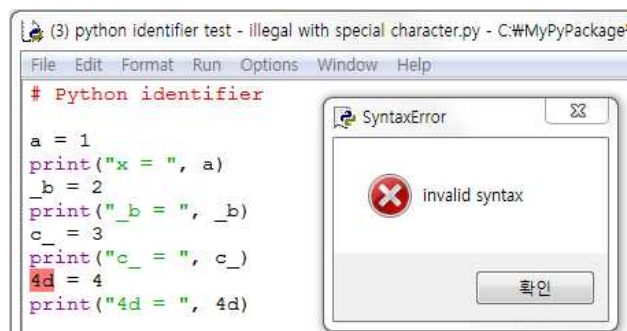
```
# Python identifier

x = int("12345")
print("x = ", x)

int = 3
y = int("6789")
print("y = ", y)
```

Python 3.7.2 Shell

Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2019)
Type "help", "copyright", "credits" or "license()" for more
>>>
RESTART: C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
t\3.1 Python identifier\2 python identifier
x = 12345
Traceback (most recent call last):
 File "C:\MyPyPackage\TextBook - 2019\ch 3
 3.1 Python identifier\2 python identifier
 y = int("6789")
TypeError: 'int' object is not callable



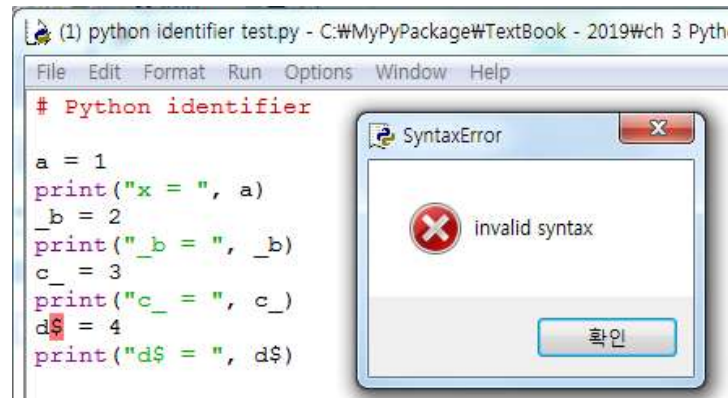
```
# Python identifier

a = 1
print("x = ", a)
_b = 2
print("_b = ", _b)
c_ = 3
print("c_ = ", c_)
4d = 4
print("4d = ", 4d)
```

SyntaxError

invalid syntax

확인



```
# Python identifier

a = 1
print("x = ", a)
_b = 2
print("_b = ", _b)
c_ = 3
print("c_ = ", c_)
d$ = 4
print("d$ = ", d$)
```

SyntaxError

invalid syntax

확인

기본 수학 연산

◆ 기본 수학 연산

파이썬 기본 연산자	예제 및 설명
+	주어진 데이터가 숫자 인 경우: 덧셈 주어진 데이터가 문자열일 경우: 문자열 병합
-	주어진 숫자 데이터에 대한 뺄셈
*	주어진 데이터가 숫자 인 경우: 곱셈 주어진 데이터가 문자열일 경우: 문자열 병합
/	주어진 숫자 데이터에 대한 실수형 나눗셈 (소수점 이하 값 유지)
//	주어진 숫자 데이터에 대한 정수형 나눗셈 (소수점 이하 값 생략)
%	주어진 숫자 데이터에 대한 모듈로 계산

함수의 호출 및 실행

◆ 파이썬 함수 호출의 예

파이썬 기본 함수 예시	예제 및 설명
<code>x_str = input("prompt string");</code>	표준 입력 장치로부터 문자열 입력 prompt 문자열이 지정되어 있으면 이를 먼저 출력하고, 문자열 입력
<code>x = int(x_str)</code>	주어진 숫자 문자열을 정수 (integer)로 변환
<code>y = float(y_str)</code>	주어진 숫자 문자열을 실수 (float)로 변환
<code>print("output string", data);</code>	<code>print("sum = ", sum)</code>

파이썬 프로그램의 입력과 출력

파이썬 프로그램의 데이터 입출력

◆ 파이썬 프로그램의 입력 및 출력 예

```
# Python input and output
```

```
a = input("input a data : ")
print("input data a = ", a)
print("type of a is ", type(a))
```

```
b = input("input an integer data : ")
print("input data b = ", b)
print("type of b is ", type(b))
sum = a + b
print("sum is : ", sum)
```

```
input a data : 10
input data a = 10
type of a is <class 'str'>
input an integer data : 20
input data b = 20
type of b is <class 'str'>
sum is : 1020
```

입력 문자열의 숫자 데이터 변환

◆ 숫자 데이터 입력

```
# Simple Python program with number input and output
```

```
a_str = input("input a_str : ")
print("input a_str = ", a_str)
print("type of a_str is ", type(a_str))
a = int(a_str) # a = int(input("input a = "))
print("type of a is ", type(a))
```

```
b_str = input("input b_str : ")
print("input b_str = ", b_str)
print("type of b_str is ", type(b_str))
b = int(b_str) # b = int(input("input b = "))
sum = a + b
print("sum is : ", sum)
```

```
input a_str : 10
input a_str = 10
type of a_str is <class 'str'>
type of a is <class 'int'>
input b_str : 20
input b_str = 20
type of b_str is <class 'str'>
sum is : 30
```



포맷 지정 데이터 출력

◆ 파이썬 프로그램의 출력 포맷 지정

Definition of Output Format	Example of Output Formatting
Output formatting with %d, %s, %f	<pre>print("%d %5d %10s, %7.3f" % (123, 456, "Python", 123.456))</pre> <p>%d : output in decimal format, right aligned %5d : output in decimal format, right aligned, in 5 character spaces %05d : output in decimal format, right aligned, in 5 character spaces, the remaining part in the left side is marked by 0's %c : output character %s : output string %10s : output left-aligned string in 10 character spaces %f : output float number %7.3f : output float number, right aligned, in 7 character spaces, decimal point and three numbers in fractional part, the remaining spaces in the fractional part is/are mark by 0's</p>
Output formatititng with {n:x} and .format method	<pre>print("{0:d} {1:5d} {2:05d} {3:10s} {4:7.3f}".format(123, 456, 678, "Python", 123.456)) print("{0:d} {1:5d} {2:05d} {3:10s} {4:7.3f}".format(123, 456, 678, "Python", 123.456))</pre> <p>{n:d} : print n-th element in decimal format {n:5d} : print n-th element in decimal format in 5 spaces, right-adjusted {n:05d} : print n-th element in decimal format in 5 spaces, right-adjusted, fill the remaining spaces with '0' {n:#07X} : print n-th element in hexadecimal format (including prefix, 0X) in 7 spaces, right-adjusted, fill the remaining spaces with '0' {n:c} : print n-th element in character format {n:s} : print n-th element in string format {n:10s} : print n-th element in string format in 10 spaces, right-adjusted {n:-10s} : print n-th element in string format in 10 spaces, left-adjusted {n:f} : print n-th element in float format {n:7.3f} : print n-th element in float format, in total 7 spaces, 3 digits of fractional part, right-adjusted, fill the remaining spaces as '0' {i:>} i-th element is right-aligned {i:^} i-th element is center-aligned {i:<} i-th element is left-aligned</p>
Output formatting with format() function	<pre>format(123456789, 'd') : insert comma (,) at every 1000's decimal unit</pre>



출력 포맷 지정 예제

◆ 출력 포맷 지정 예제

```
# Python print() with formatting

print("%5d %7d %07d %10s %7.3f" % (123, 456, 678, "Python", 123.456789))
print("%5d %7d %07d %-10s %10.6f" % (12, 34, 5, "Python", 123.456789))

print("{0:5d} {1:7d} {2:07d} {3:10s} {4:7.3f}" \
      .format(123, 456, 678, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:<10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:^10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:>10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("Long integer with comma at each 1000 unit : " \
      , format(123456789, ',d'))
```

```
123      456 0000678      Python 123.457
12       34 0000005 Python    123.456789
123      456 0000678 Python    123.457
12       34 0000005 Python    123.456789
12       34 0000005 Python    123.456789
12       34 0000005 Python    123.456789
12       34 0000005 Python    123.456789
Long integer with comma at each 1000 unit : 123,456,789
```



한 줄에 여러 정수 (integer) 데이터 입력 - split(), map()

```
# input multiple numbers from one line using split() and map()
```

```
input_data_str = input("input data : ")
decimal_strings = input_data_str.split(sep=' ')
print("Input decimal_strings : ", decimal_strings)
L = list(map(int, decimal_strings))
print("Input integers : ", L)
```

```
input data : 1 2 3 4 5 6 7 8 9 10
Input decimal_strings : ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
Input integers : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
input data : 1 1234 6543 9876
Input decimal_strings : ['1', '1234', '6543', '9876']
Input integers : [1, 1234, 6543, 9876]
```



한 줄에 여러 실수 (float) 데이터 입력 - split(), map()

```
# Getting multiple float numbers in one line using split() and map()
```

```
input_data_str = input("input float data : ")
float_strings = input_data_str.split(sep=' ')
print("Input decimal_strings : ", float_strings)
L = list(map(float, float_strings))
print("Input float data : ", L)
```

```
input float data : 1.23 4.56 7.78 10.12345
Input decimal_strings : ['1.23', '4.56', '7.78', '10.12345']
Input float data : [1.23, 4.56, 7.78, 10.12345]
```

파이썬 프로그램 실행 제어 기초

- 조건문과 반복문 개요

조건문 개요 – if, if-else

◆ 조건문 if

```
# simple program to input two integers and compare
x = int(input("input first integer (x) : "))
y = int(input("input second integer (y) : "))
if x == y:
    print("x(%d) is equal to y(%d)"%(x, y))
if x < y:
    print("x(%d) is less than y(%d)"%(x, y))
if x > y:
    print("x(%d) is greater than y(%d)"%(x, y))
```

```
input first integer (x) : 7
input second integer (y) : 5
x(7) is greater than y(5)
```

```
input first integer (x) : 3
input second integer (y) : 3
x(3) is equal to y(3)
```

```
input first integer (x) : 4
input second integer (y) : 8
x(4) is less than y(8)
```

조건문 – if-elif-else

◆ 조건문 if-elif-else

```
# conditional branch with if - elif - else
score = int(input('course score [0..99] = '))
if 90 <= score <= 100:
    grade = 'A'
elif 80 <= score:
    grade = 'B'
elif 70 <= score:
    grade = 'C'
elif 60 <= score:
    grade = 'D'
else:
    grade = 'F'
print("score = %d, grade = %s" %(score, grade))
```

```
course score [0..99] = 95
score = 95, grade = A
```

```
course score [0..99] = 74
score = 74, grade = C
```



while 반복문 개요

◆ while-loop 기본 구조

```
# while-loop for input positive integers

L = list() # creation of an empty list L
print("Input positive integers (-1 to end)")
x = int(input("data : "))
n = 0
sum = 0
while x >= 0:
    L.append(x) # append x to list L
    n += 1
    sum = sum + x
    x = int(input("data : "))

print("Total", n, " integers were input: ", L)
print("Sum is : ", sum)
```

```
Input positive integers (-1 to end)
data : 1
data : 3
data : 5
data : 7
data : 9
data : 2
data : 4
data : 6
data : 8
data : 10
data : -1
Total 10 integers were input: [1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
Sum is : 55
```


while-loop과 continue, break

◆ while-loop with break and continue

```
# while-loop, break, continue

MAX_NUM_DATA = 100
num_data = 0
data_sum = 0
print("Input (up to {} ) integer data.".format(MAX_NUM_DATA))
while num_data < MAX_NUM_DATA:
    data = int(input("Data (-1 to finish) = "))
    num_data = num_data + 1
    if data == -1:
        break
    elif data <= 0:
        continue
    else:
        data_sum = data_sum + data

print("Total {} data input, sum of positive data = {}".format(num_data, data_sum))
```

```
Input (up to 100 ) integer data.
Data (-1 to finish) = 10
Data (-1 to finish) = 2
Data (-1 to finish) = 7
Data (-1 to finish) = 1
Data (-1 to finish) = -5
Data (-1 to finish) = -9
Data (-1 to finish) = 2
Data (-1 to finish) = 3
Data (-1 to finish) = 13
Data (-1 to finish) = -1
Total 10 data input, sum of positive data = 38
```



입력 데이터 중의 최댓값, 최솟값 찾기

◆ Find min, max of input data list

```
# while-loop, find min and max of input data list

TARGET_NUM_DATA = 10
num_data = 0
L_data = [] # empty list
L_sum = 0
print("Input up to {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    num_data = num_data + 1
    L_data.append(data)
    L_sum = L_sum + data
    if num_data == 1:
        data_min = data_max = data
        continue
    if data < data_min:
        data_min = data
    if data > data_max:
        data_max = data

print("Input data list = ", L_data)
print("Min = {}, Max = {}, Avg = {}".\
      .format(data_min, data_max, L_sum/num_data))
```

```
Input up to 10 integer data.
data = 32
data = 45
data = 1
data = 5
data = 10
data = 4
data = 3
data = 99
data = -5
data = -30
Input data list = [32, 45, 1, 5, 10, 4, 3, 99, -5, -30]
Min = -30, Max = 99, Avg = 16.4
```



range() Generator 함수

◆ range(start, stop, step)

- start로 부터 stop 직전까지 (stop은 포함하지 않음) step씩 증가하며 정수 값을 생성하여 제공
- 만약 인수가 2개만 제공되면, start와 stop의 값으로 사용되며, step은 1로 설정됨
- 만약 인수가 1개만 제공되면 stop의 값으로 사용되며, start는 0으로, step은 1로 설정됨
- 만약 step 값이 음수이면, start로 부터 stop 직전까지, step에 따라 감소시키며 정수값을 생성하여 제공

range() 사용 예	설 명
range(10)	0부터 9까지 1씩 증가하며 정수를 생성
range(1, 10)	1부터 9까지 1씩 증가하며 정수를 생성
range(1, 10, 2)	1부터 9까지 2씩 증가하며 정수를 생성
range(10, 100, 5)	10부터 99까지 5씩 증가하며 정수를 생성
range(100, 0, -1)	100부터 1까지 1씩 감소하며 정수를 생성

range()와 in을 사용하는 for 반복문

◆ for-loop

- for-loop에서는 in과 range() 를 사용하여 구성할 수 있음
- for-loop에서 사용하는 변수 (아래 예에서는 i)의 값이 range()에서 생성된 값에 차례로 대입되어 사용되며, 전체 반복 회수는 range()에서 생성된 숫자의 개수에 의해 제한됨

```
# for-loop with range()
n = int(input('Input n to calculate sum of [0..n] : '))
nSum = 0
for i in range(0, n+1): #nSum = sum(range(0, n+1))
    nSum = nSum + i
print("Sum of [0..%d] = %d" %(n, nSum))
```

```
Input n to calculate sum of [0..n] : 10
Sum of [0..10] = 55
```

```
Input n to calculate sum of [0..n] : 100
Sum of [0..100] = 5050
```



for 반복문과 continue, break

◆ Example of for-loop with break and continue

```
#for_loop with break and continue

n = int(input("Input number of data to process: "))
L = list()
sum = 0
print("Input %d non-negative integers"%(n))
for i in range(n):
    d = int(input())
    if d == 0:
        continue
    elif d < 0:
        break
    L.append(d)
    sum = sum + d
print("Input data : ", L)
print("Sum = ", sum)
```

```
===== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
3
0
4
5
Input data : [2, 3, 4, 5]
Sum = 14
>>>
===== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
-1
Input data : [2]
Sum = 2
>>> |
```

입력데이터의 통계 분석 – 평균, 분산, 표준편차

```
# while-loop, for-loop, find min and max of input data list
import math
```

```
TARGET_NUM_DATA = 10
num_data = 0
L_data = [] # empty list
L_sum = 0
print("Input {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    num_data = num_data + 1
    L_data.append(data)
    L_sum = L_sum + data

avg = L_sum / num_data
diff_sq_sum = 0
for i in range(num_data):
    diff = avg - L_data[i] # difference from avg
    diff_sq_sum = diff_sq_sum + diff * diff
var = diff_sq_sum / num_data
std = math.sqrt(var)

print("Input data list = ", L_data)
print("avg = {}, var = {}, std = {}".format(avg, var, std))
```

```
Input 10 integer data.
```

```
data = 1
data = 2
data = 3
data = 4
data = 5
data = 6
data = 7
data = 8
data = 9
data = 10
```

```
Input data list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
avg = 5.5, var = 8.25, std = 2.8722813232690143
```

```
Input 10 integer data.
```

```
data = -5
data = -4
data = -3
data = -2
data = -1
data = 0
data = 1
data = 2
data = 3
data = 4
```

```
Input data list = [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]
avg = -0.5, var = 8.25, std = 2.8722813232690143
```



ASCII Characters

◆ ASCII (American Standard Code for Information Interchange)

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0x10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x20	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0x30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x50	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0x60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Printing Upper-case/Lower-case Characters and Numbers in ASCII table

```
# Printing Upper_case / Lower_case Characters, and Digits of ASCII Code Table
```

```
L_upper = [] # list
for x in range(0x41, 0x5B):
    L_upper.append(chr(x))
print("Upper case alphabets : \n", L_upper)
print()
```

```
L_lower = []
for x in range(0x61, 0x7B):
    L_lower.append(chr(x))
print("Lower case alphabets : \n", L_lower)
print()
```

```
L_digits = []
for x in range(0x30, 0x3A):
    L_digits.append(chr(x))
print("Digits : \n", L_digits)
print()
```

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0x10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x20	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0x30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x50	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0x60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

```
Upper case alphabets :
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

Lower case alphabets :
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']

Digits :
['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
```


객체 지향형 프로그래밍 개요

객체 지향형 프로그래밍 (Object-Oriented Programming)

◆ 객체 (Object)

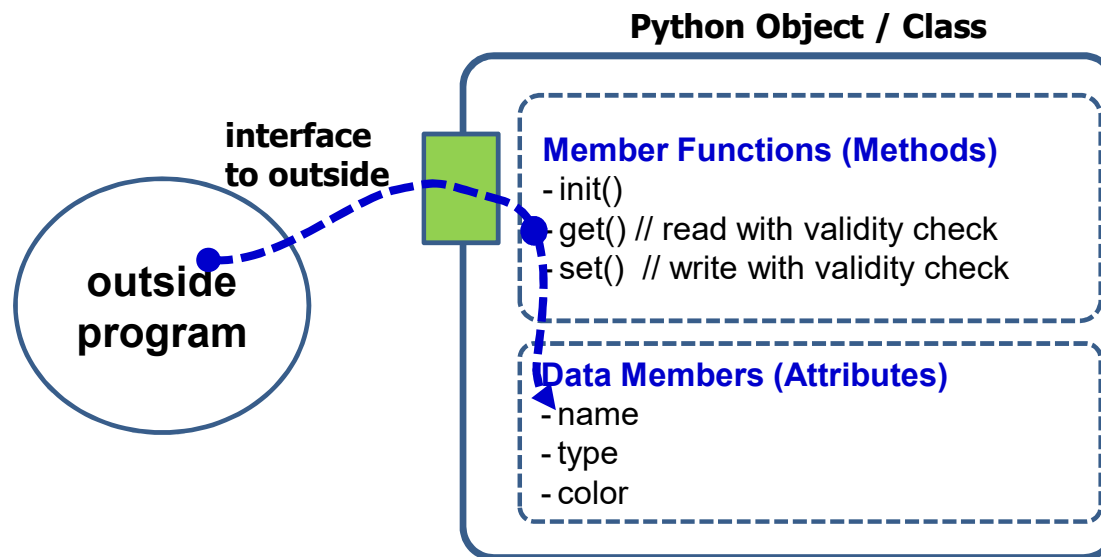
- 우리 주변에서 살펴볼 수 있는 다양한 사물과 생물들을 의미하며 크기, 색깔, 위치 등의 상태 (state) 정보와 스스로 또는 다른 객체에 의하여 이루어지는 행동 (behavior)을 가짐.
- 예를 들어 사람에게서는 이름, 생년월일, 키, 몸무게, 현재 위치, 소속 기관 등의 상태 정보와 그 사람이 다른 곳으로 이동하거나, 키와 몸무게가 변하거나, 노래를 부르거나, 소속기관이 변경되는 등의 행동이 이루어질 수 있음.

◆ 객체 지향형 프로그래밍

- 소프트웨어 시스템을 객체 (object) 단위로 나누고, 각 객체는 그 내부에 데이터와 관련 함수를 함께 포함하게 함으로써 독립성을 유지하게 하는 프로그래밍 기법
- 객체 지향형 프로그래밍에서는 문제를 해결하는 함수 호출의 절차적 실행 순서보다는 각 객체가 어떻게 상호 연관성을 가지며, 각 객체 내부에서 데이터를 어떻게 정상적인 범위 내에 유지하게 하고, 그 데이터를 어떤 자료구조를 사용하여 가장 효율적으로 표현하며, 어떤 알고리즘을 사용하여 가장 효율적으로 처리할 수 있는가에 대하여 중점을 두게 됨.

파이썬 클래스

◆ 클래스 (class)



클래스와 인스턴스

◆ 클래스 (class)

- 객체를 소프트웨어에서 표현하고 구현하기 위하여 사용하는 틀/모델
- 클래스는 객체 지향형 프로그래밍으로 알고리즘 및 자료구조를 구현하기 위한 기본 단위

◆ 인스턴스 (instance)

- 클래스를 사용하여 실체를 만든 것
- 자료형 (예: int, float)을 사용하여 변수 (예: x, y)를 만드는 것과 같이 클래스 (예: list)를 사용하여 다수의 인스턴스 (예: lst_name, lst_value)를 생성할 수 있음

파이썬 클래스

◆ 클래스의 기본 멤버 함수

클래스의 기본 멤버 함수	설 명
생성자 (constructor)	클래스를 사용하여 객체가 생성될 때 실행되며, 초기화 기능 수행
데이터 멤버 접근자 (accessor)	클래스의 데이터 멤버 값을 읽기 위하여 접근하는 기능 제공. 예) getXXX()
데이터 멤버 변경자 (mutator)	클래스의 데이터 멤버 값을 변경하기 위한 기능 제공. 예) setXXX()

◆ dot (.) 연산자

- 클래스나 객체의 멤버 데이터와 멤버 함수를 사용할 때 dot(.) 연산자를 사용하여 멤버 관계를 나타냄

속성 (attribute)과 메소드 (method)

◆ 속성 (attribute)

- 클래스 (class)와 클래스로 부터 생성된 인스턴스(instance)가 가지는 멤버 데이터 또는 멤버 함수
- 클래스가 구현하는 기능과 정보를 의미
- 클래스/인스턴스의 이름에 dot(.)을 붙이고 속성을 지정
- 클래스/인스턴스의 속성 설정/변경 및 접근을 위한 멤버함수 (메소드)를 클래스가 제공
- 예)
t = turtle.Turtle()
t.color('red')

◆ 메소드 (method)

- 클래스와 인스턴스가 제공하는 멤버 함수
- 클래스/인스턴스의 이름에 dot(.)을 붙이고 메소드를 지정
- 예)
t.forward(100)

파이썬과 객체지향형 프로그래밍

◆ 파이썬은 객체지향형 프로그래밍 기반

- 파이썬에서 사용되는 모든 자료형, 모듈 및 패키지는 객체지향형으로 구현되어 있음
- 따라서 자료형, 모듈 및 패키지에 dot(.) 연산자를 사용하여 속성(멤버 함수와 멤버 데이터)를 지정하게 됨
- 가장 기본적인 객체: PyObject (파이썬 객체)
- 객체 지향형 프로그래밍을 사용함으로써 모든 자료형을 하나의 체계로 관리할 수 있고, 동적 자료형 (dynamic typing)이 가능함

터틀 그래픽 예제

터틀 그래픽의 기본 함수

터틀 그래픽 기본 함수	설 명
Turtle()	터틀 객체의 생성
window_width()	윈도우 넓이 읽기
window_height()	윈도우 높이 읽기
setup(width, height)	캔버스의 크기를 가로 (width)와 세로(height)로 설정
shape()	터틀 객체의 모양을 설정 (classic: 화살 축, circle: 원, square: 사각형, triangle: 삼각형, arrow: 화살 축)
shapeseize()	터틀의 크기를 설정
bgcolor(color)	배경색을 설정
circle(radius, steps)	주어진 반지름 (radius)의 원을 그림, steps가 주어지면 다각형을 그림
color(color)	터틀 객체의 색상을 설정
dot(size)	size 크기의 점을 그림
goto(x_coord, y_coord)	x좌표와 y좌표로 이동
forward(distance)	지정된 거리만큼 현재의 방향으로 전진
backward(distance)	지정된 거리만큼 현재의 반대 방향으로 후진
left(angle)	왼쪽 (반 시계방향)으로 지정된 각도만큼 회전
right(angle)	오른쪽 (시계방향)으로 지정된 각도만큼 회전
penup()	터틀의 펜을 위로 들어 이동에 따른 그리기가 나타나지 않게 함
pendown()	터틀의 펜을 아래로 내려 이동에 따라 그리기가 나타나게 함
setposition(x, y)	터틀의 위치를 지정된 좌표(x, y)로 설정
speed()	터틀 이동 속도를 설정: 1:가장느림~9:빠름, 0: 가장빠름
textinput()	입력창을 사용하여 문자열 입력
write(label)	현재의 터틀 위치에 label 문자열을 출력



터틀 그래픽 - 4각형 그리기

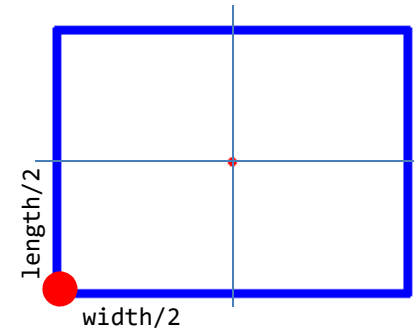
```
# Drawing Rectangle with inputs for width and length
import turtle

turtle.setup(500, 500) # window width, height
turtle.title("Drawing a rectangle")
t = turtle.Turtle()
t.shape("turtle")
t.pensize(5)
t.color('blue')
t.pencolor('red')

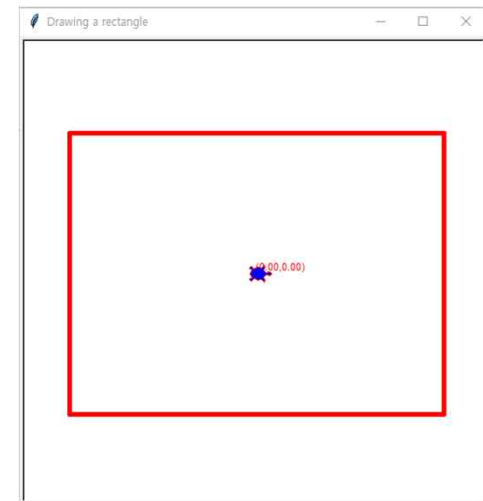
width, length = map(int, input("width and length = ").split())
print("Drawing a rectangle of width({}), length({}) ....".format(width, length))

t.dot(3)
t.write(t.pos())
t.up(); t.goto((-width/2, -length/2)); t.down()

t.forward(width); t.left(90)
t.forward(length); t.left(90)
t.forward(width); t.left(90)
t.forward(length); t.left(90)
t.up(); t.goto((0, 0)); t.down()
```



width and length = 400 300
Drawing a rectangle of width(400), length(300)



터틀 그래픽 – 3각형 그리기

```
# Drawing a triangle with for-loop
import turtle
import math

turtle.setup(500, 500) # window width, height
turtle.title("Drawing a triangle")
t = turtle.Turtle()
t.shape("turtle")
t.pensize(5)
t.color('blue')
t.pencolor('red')

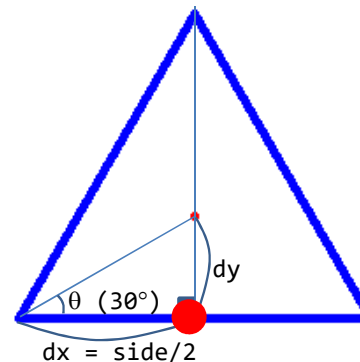
side_length = int(input("side_length = "))
print("Drawing a triangle of side_length({}) ....".format(side_length))

t.dot(3)
t.write(t.pos())
dx = side_length / 2
dy = side_length * math.tan(math.pi/6) / 2
turn_angle = 360 / 3
t.up(); t.goto((-dx, -dy)); t.down()

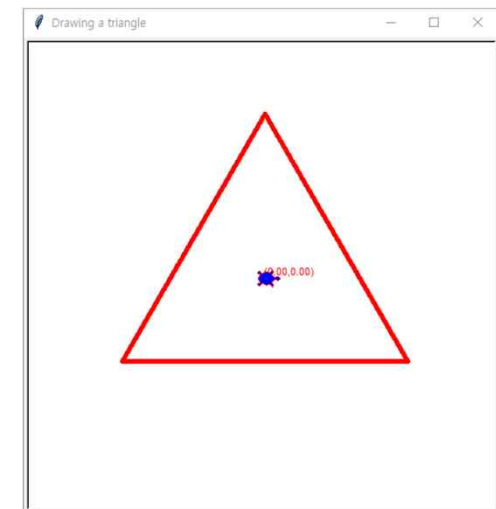
for i in range(3):
    t.forward(side_length)
    t.left(turn_angle)

t.up(); t.goto((0, 0)); t.down()
```

side_length = 300
Drawing a triangle of side_length(300)



$dx = side / 2$
 $\tan(\theta) = 2.0 * dy / side$
 $dy = side * \tan(\theta) / 2.0$



별 그리기

Simple Python Program to Draw a Star with given center position
import turtle, math

```

turtle.setup(500, 500) #set width and height of canvas
turtle.title("Drawing a star at given position")
t = turtle.Turtle()
t.shape('classic')

```

```

num_vertices = 5
center_x, center_y = map(int, input("input center_x and center_y : ").split(' '))
center = (center_x, center_y)
line_length = int(input("line_length of star : "))
line_color = input("line_color of star (e.g., red, blue) = ")
t.pencolor(line_color)

```

```

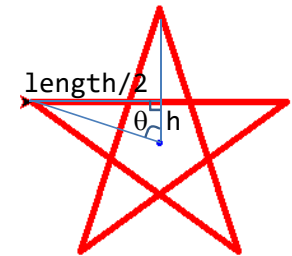
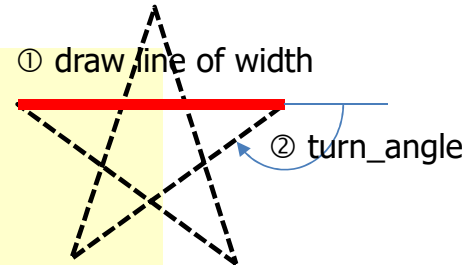
t.dot(10, "red"); t.write(t.pos())
t.up(); t.goto(center); t.down()
t.dot(10, "blue"); t.write(center)
start_x = center_x - line_length/2
theta = math.radians(360 / num_vertices) # convert angle in degree into angle in radian
h = line_length / (2 * math.tan(theta))
start_y = center_y + h
t.width(5)
t.penup(); t.goto(start_x, start_y); t.dot(10, "blue")
t.write((start_x, start_y)); t.pendown() #pen down to draw

```

```

for i in range(num_vertices):
    t.forward(line_length)
    t.right(2*360/num_vertices)
    t.up(); t.goto(center); t.down()

```



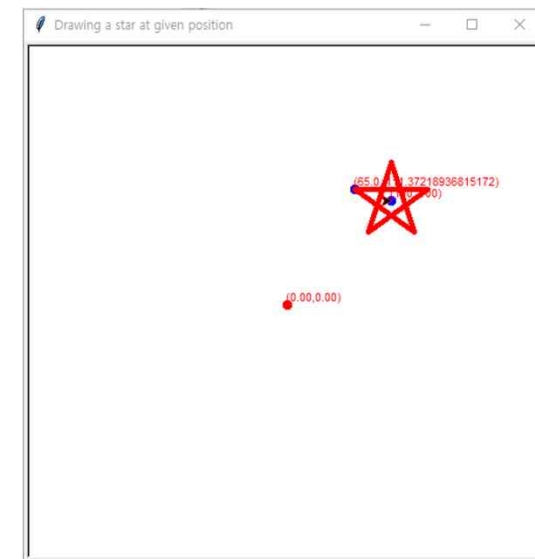
$$\tan(\theta) = \text{length} / (2.0 * h)$$

$$h = \text{length} / (2.0 * \tan(\theta))$$

```

input center_x and center_y : 100 100
line_length of star : 70
line_color of star (e.g., red, blue) = red

```



Homework 1

Homework 1

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

- 1.1 표준입력장치 (키보드)로부터 원의 반지름(radius)을 입력 받고, 그 원의 넓이(area)와 원둘레 (circumference)를 출력하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.
(실행 예제)

```
radius = 10  
Circle of radius (10) : area (314.1592), circumference(62.83184)
```

- 1.2 직사각형의 가로 (width)와 세로 (length)를 입력 받아 넓이(area)와 둘레(perimeter)를 계산하여 출력 하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.
(실행 예제)

```
width, length = 100 50  
Rectangle of width(100) and length(50) : area (5000), perimeter(300.0)
```



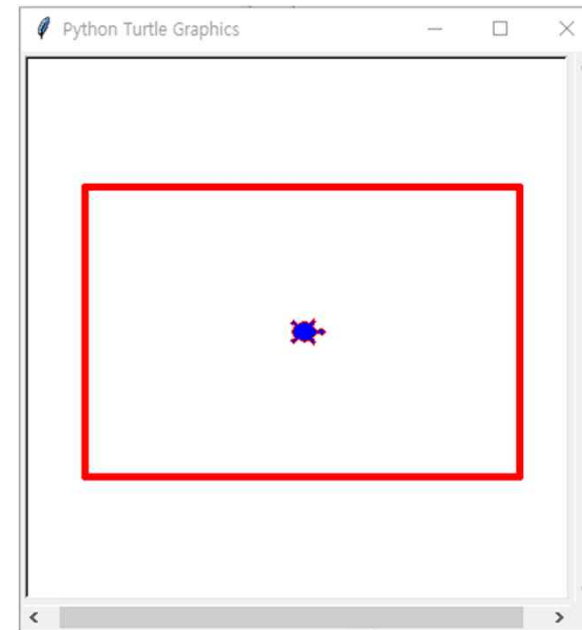
Homework 1

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

1.3 표준입력장치 (키보드)로부터 직사각형의 가로 (width) 및 세로 (length) 크기를 각각 입력 받고, 터틀 그래픽을 사용하여 지정된 크기의 사각형을 (0, 0) 좌표가 중심이 되도록 그리는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.

(실행 예제)

```
width = 300
length = 200
Drawing a rectangle of width(300), length(200) ....
```



References

- [1] 김영탁, **컴퓨팅 사고와 파이썬 프로그래밍**, 홍릉과학출판사, 2022. 1.
- [2] 김영탁, **자료구조와 알고리즘을 함께 배우는 파이썬 프로그래밍**, 홍릉과학출판사, 2021. 8.
- [3] Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, Data Structures and Algorithms in Python, Wiley, 2013.
- [4] Kent D. Lee, Steve Hubbard, Data Structures and Algorithms with Python, Springer, 2015.
- [5] Nikhil Ketakar, Deep Learning with Python, Apress, 2017.
- [6] Antonio Gulli, Amita Kappor, Sujit Pal, Deep Learning with TensorFlow 2 and Keras, Packt, 2019.
- [7] 김동근, 쉽게 배우는 파이썬 프로그래밍, 가메출판사, 2016.
- [8] 천인국, 파워유저를 위한 파이썬 Express, 생능출판사, 2020. 11.
- [9] **Python Software Foundation**, <https://www.python.org/>.
- [10] Python Tutorial, <https://docs.python.org/3/tutorial/>.
- [11] Summary **of Turtle Methods**, <http://interactivepython.org/runestone/static/IntroPythonTurtles/Summary/summary.html>.
- [12] 김영탁, 자료구조와 알고리즘을 함께 배우는 C 프로그래밍, 배움터, 2020. 2.
- [13] 김영탁, 자료구조와 알고리즘을 함께 배우는 C++ 프로그래밍, 배움터, 2020. 8.

