11. microsoft ai

Ai를쓰는 이유

1. Big data
2. 문제해결 할 것이 많아 짐 -> 컴퓨터 자원이 많이 필요
3. 최신형의 알고리즘의 필요성 강조

Ai의 역사

1970년대부터 논문서 등장하기 시작하였지만, 여러가지 문제가 있었음.

1990년대부터 근본적인 문제가 해결되면서 성장이 빨라짐.

Ai의 분야

1. vision
2. speech
3. language
4. knowledge

마이크로소프트가 ai를 활용하는 방법

* ai의 민주화

1. agent: 사람들의 옆에서 쉽게 사용할 수 있도록 제공
2. application: office365, teams, excel, ppt, visual studio
3. service: bot framework(대화형 챗봇), 인지 서비스(자연어 처리, 딥러닝 기능), cognitive toolkit(딥러닝 패키지)
4. infrastructure

파이썬을 활용한 데이터 분석

1. python소개와 개발환경, 코딩 규칙

특징

:쉬운 문법

:간결함

:확장성이 강함

:오픈소스

:개발 속도 빠름(프로토타입 구현, 전체 개발 시간 단축)

사용 사례

1. 시스템 유틸리티
2. Gui 프로그래밍
3. 웹프로그래밍(많은 라이브러리)
4. c/c++과의 결합
5. 수치 연산 및 프로그래밍
6. 데이터베이스 프로그래밍
7. 적합하지 않은 용도(하드웨어 제어, 빠른 연산을 요구하는 문제)

파이썬 주의 사항

1. 들여쓰기 가증(공백 4칸)
2. 한 줄에 79자 이하(여러 줄은 괄호 사용)
3. import하나 당 하나의 라이브러리(표준, 서드 파티, 로컬 애플리케이션, 라이브러리 순으로)
4. 변수 명에 가독성이 안 좋은 기호는 넣으면 안됨
5. 한 줄/ 여러 줄 주석 모두 # 사용

클래스, 함수 변수의 이름 명명 법

1. 클래스와 그 외는 camelcase표기법 사용
2. 함수와 변수 명에는 소문자, \_사용
3. 전역 상수는 대문자와 \_ 사용

일반적인 오류를 보는 법

1. 터미널에서 어떤 스크립트를 실행했는가
2. Python이 몇 째 줄에 오류가 있는지 알려줌
3. ^ 캐럿 문자가 문제 지점 표시
4. Syntaxerror로 오류 생긴 원인 표시(오류 내용을 검색 엔진으로 검색)
5. 변수의 개념과 데이터 형식

기본 데이터 형식

1. 프로그램 구성요소(데이터+연산)
2. 데이터 형식: 비슷한 특징을 가지는 데이터들의 유형
3. 가변형, 불변형
4. 단일 데이터 리터럴(Boolean,numeric(int, float, 허수 complex)
5. 컨테이너 형식 리터럴(숫자,문자,문자열 등 변화가 없는 값인 상속값에 대한 데이터 표기)

* 시퀸스: 순서가 있는 데이터들의 모음
* 텍스트 시퀸스: 문자열
* 바이너리 시퀸스: 바이트
* 집합: 비 시퀸스 데이터 형식, 가변형, 중복 허용 안함
* 매핑: 딕셔너리

시퀸스

1. 데이터 요소가 순서를 가짐.
2. List

* 가변형, 값 중복 허용
* 타 데이터 형식 리터럴 포함 가능
* 다른 리스트, 튜플, 딕셔너리 요소

1. Tupe

* 불변형, 값 중복 허용
* Enumerate,zip,분산 처리 참조 값등에 사용
* ,를 통해 분리/괄호 안에 ,를 통해 작성

1. Range

* 순차적인 정수 리스트 생성
* 범위의 마지막은 포함 안됨
* 안의 속성은 보여주지 않고 전체적인 범위값을 보여줌

텍스트 시퀸스

1. 불변형
2. 유니코드 문자열 다룰시:utf-8로 인코딩 -> 바이너리 시퀸스
3. 데이터 리터럴 만드는 방법(‘,”-한 줄 문자열 작성시,’’’/”””-여러줄 문자열 작성시)

바이너리 시퀸스

1. 인코딩한 데이터: 문자열 앞에 b(바이너리 시퀸스)작성
2. 바이트 타입: 바이트-불변형

: 바이트 배열- 가변형

집합( set)

1. 수학 집합 연산 메소드 제공: union(합집합), intersection(교집합), difference(차집합), symmetric difference(대칭차)
2. 리스트, 튜플의 데이터 중복 제거
3. 데이터 모음의 요소 유무 검사의 성능이 우수
4. 숫자, 문자열, 바이트 범위, 튜플
5. 리스트, 딕셔너리, 바이트 어레이는 사용 못함.

매핑(딕셔너리)

1. 해시 알고리즘 사용(집합과 동일)
2. 키와 값 쌍

* 값은 리스트 형식으로 데이터 변경 가능
* 키는 숫자, 문자 튜플 사용, 리스트 사용 불가

변수 개요

: 데이터를 저장하는 틀

: 데이터가 저장되는 메모리 위치를 가리키는 이름

: 변수 선언 형식- 변수=데이터

* 설명적인 변수 이름 사용

문자열과 변수

1. 따옴표 안에 든 데이터 -> 문자열

: ‘ or “

1. 변수가 담긴 문자열 만들기

: f”문자열(변수)

: 소문자 f(format)

1. 미리 만들어 놓은 문자열: 포멧

: .format(변수/문자열/숫자/boolean)

입력과 변수

1. 입력으로 데이터 저장하기

: 변수 = input()

1. 입력에 프롬프트 표시하기

: 변수 = input(“프롬프트”)

: 줄 바꿈 문자 제거- end=’ ’

스크립트와 변수

1. 스크립트에 변수 전달하기

: python test.py

:실행 인자

: 실행 인자 변수 – argv

: 실행 인자를 변수에 따로 담기 = unpacl

1. 기능 가져오기

: import 모듈이름

: from 모듈이름 import 모듈이름

1. Python 연산, 입력과 출력을 처리하는 방법

Python의 기본 연산자

1. 연산을 수행하는 도구
2. 연산의 주요 용어

: 피연산자(숫자)

: 연산자(연산 부호)

: 연산(전체 식)

1. 연산의 종류

: 수학 연산

: 논리 연산(참.거짓)

1. 연산자의 종류

: 단항 연산자(+/-, ++/--, !)

: 이항 연산자 – 대부분의 연산자

: 삼항 연산자 -> [결과(조건 참)] if [조건] else [결과( 조건 거짓)]

주요 연산자

1. 산술 연산자: +,-,/,\*,%
2. 논리 연산자: <,>,<=,>=
3. 식별 연산자: is, is not
4. 연산자의 우선순위: (), 지수, 곱셈과 나눗셈, 덧셈과 뺄셈
5. 괄호의 목적

: 연산자의 기본 우선순위 변경

: 코드의 가독성 및 명확성 향상

: bool b = a % 3 > c / 4 -> boo b = (a%3)>(c/4)

논리 연산

1. 연산자와 피연산자의 결합이 어떤 지점에서 참/거짓 판별
2. 논리기호/연산자

: and, or, not

: !=, ==

: <,>,<=,>=

: True, False

1. 논리 구문 푸는 단계
2. 등가 검사(!=, ==)
3. 괄호 내의 and/or
4. Not 적용
5. 남은 and/or 결정

Print 함수 기본 사용법

1. Python의 표준 출력 함수
2. Print(value1, value2…)
3. 출력 값 사이 구분자 지정, sep

: 기본값- 공백

1. 출력 마지막 처리 지정, end

: 기본값- 줄 바꿈

문자열 출력 총 정리

1. 작은 따옴표나 큰 따옴표 사용

: ‘ “ ‘/ “ ‘ “

1. 확장 문자 \가 필요한 경우

: “ “ “/ ‘ ‘ ‘

1. 동일 문자열 여러 번 출력

: “value \* 횟수”

확장 문자(escape sequence)

1. 입력하기 어려운 문자 입력시
2. 지원하는 확장 문자 목록

: \n – 줄 바꿈, \t – 탭, \\ - 문자\ 출력, \’ – ‘ 출력, \”-“출력, \r-현재 커서를 가장 앞으로 이동, \f-현재 커서를 다음줄로 이동, \a-삑 소리 남, \b – 백 스페이스, \000-널문자

Input 함수 기본 사용법

1. Python의 표준 입력 함수
2. Input()
3. Input(‘문자열’)
4. 사용자 eof -> eoferror

: 리눅스 – ctrl+d

: window – ctrl+z+return

정수 입력

1. 문자열 입력을 정수형으로 변환
2. Eval()함수: python 표현식으로 반환
3. Int() 클래스

: 숫자나 문자열에서 정수 객체 반환

: 인자가 없으면 0 출력

: 진법 지정 가능(기본 10진법) -> int(str, base)

실수 입력

1. 문자열 입력을 실수로 변환
2. Eval(str) 함수: python 표현식으로 반환
3. Float(str) 클래스

: 실수 객체 반환

: 인자 없으면 0 반환

튜플과 리스트로 입력

1. 문자열 입력을 튜플과 리스트로 변환
2. Eval(str) 함수: python 표현식으로 반환

Argv와 input응용

1. 프롬프트 변수로 반복 줄이기
2. 게임 실행 방식
3. 2종류 입력 함께 쓰기

: 스크립트 실행시 입력 – argv

: 스크립트 실향중 입력 – input

1. 파이썬 파일처리

파일 기본

1. 파일 객체

: 파일의 내용을 읽는 수단

:파일 포인터가 읽는 위치

1. 파일 객체 열기

: open

:pydoc open

1. open함수의 인자

: file -> 필수값, 파일 경로

: mode -> r,w,x,a,b,t,+,U

: buffering -> 0,1 보다 큰 양수

:encoding -> utf8명시

1. 파일 객체 닫기

: close()

: 닫지 않으면 버퍼일 된 데이터 소실

파일 읽기

1. Read(): 파일의 전체 내용 반환
2. Read(글자수): 글자수 만큼 읽고 파일 포인터를 끝으로 이동, 파일을 끝까지 읽은후 빈문자 반환
3. Seek(포인터 위치)

: 포인터를 원하는 위치로 이동

: 맨앞으로 이동은 seek(0)

1. Readline(): 파일에서 한 줄씩 읽기
2. Readlines(): 한줄씩 읽고 리스트 형식으로 반환

파일 쓰기

1. 파일을 쓰기로 오픈

: mode = wt(쓰기 텍스트 보드)

: mode = at(추가 텍스트 모드)

1. Write(‘내용’)
2. Write(‘내용\n’)
3. Writelines([‘내용1’,’내용2’,…])
4. Truncate(): 파일 내용 지우기

Finally 와 파일 구문으로 파일 다루기

1. Try~finally 구문: 문제가 발생해도 finally절은 실행

Try:

변수=open(파일 경로,옵션)

파일조작

Finally:

변수.close()

1. With 블록: 명시적 close() 메서드 호출이 필요 없음

With open(파일경로) as 변수:

파일 객체 처리

응용- 파일복사

1. From os.path import exists
2. Len(입력\_자료): 문자열 길이를 숫자로 반환
3. Open(원본파일).read(): close()호출 필요 없음
4. 문자열을 따옴표로 제대로 마치지 못한 경우

: EOL while scanning string literal error

1. python함수의 이해

함수 개요

1. 구조적/절차적 프로그래밍 용어
2. 여러구문을 하나로 묶은 실행 단위: 기능의 재사용 촉진
3. Python 함수의 3가지 특징
4. 코드 조각에 이름 부여
5. 실행 인자를 받을 수 있음
6. 작은 스크립트나 작은 명령 만듬
7. Python 함수의 구조

:return운 생략 가능

: 파라미터로 \*args 사용해 argv 효과 줌

Def 함수\_이름(파라미터):

실행 코드

(옵션)return 결과 값

함수 인자

1. 기본 값 인자

: def function(a, b=value):

: 인자를 입력하지 않으면 기본 값 적용

1. 위치 인자

: def function(a,b,c,…)

: 함수에서 정의한 대로 적용

1. 키워드 인자

: 키워드를 이용해 순서 무시 입력

: 위치 인자와 키워드 인자 혼합(위치가 우선)

함수와 변수

1. 함수에 실행인자를 전달하는 방법

: 값 그 자체

: 변수

: 계산

1. 함수 내의 변수와 스크립트 변수

: 변수의 생존 범위 다름

: 함수 밖에서 함수로 전달된 변수는 임시 변수로 동작후 소멸

: 함수 변수와 전역 변수는 다른 이름으로 지정

1. 함수 내에서 함수 실행 가능
2. 함수에 전달하는 실행인자의 수: 최대 5개

함수와 파일

1. python에서 모든 타입은 객체
2. 함수에 파일 객체 전달

: 함수 호출전 open()함수 호출

: 함수 호출 후 close() 함수 호출

1. 전달된 파일 객체의 읽기와 쓰기

: 파일 객체의 멤버 함수 이용

: read(), readline(), readlines()

: write(), writelines()

: seek()

반환하는 함수

1. 함수가 반환하는 값을 변수로 받기
2. Return 문

: 결과 값 반환’

: 함수 종료

1. 함수 실행 인자에 다른 함수 반환 값 사용
2. Python의 선택문과 반복문

If 절

1. 구문 형식

:if[조건식]:

: 조건식의 결과는 참 혹은 거짓

1. 조건식의 유형

: 관계 연산자로 구성된 식

: 문자열 조건

Else롸 if

1. 형식1

If(조건식):

구문…

Else:

1. 형식2

If(조건식):

구문…

Elif:

구문…

Else:

구문…

1. Elif

: 다른 언어의 경우, else if

: 대안실행을 위한 평가

중첩 if 문

: 한 분기에서 다른 분기로 가지치기

: 선택지가 더 필요하면, elif 추가

If문 모범 사례

1. 모든 if 문에는 else를 반드시 씀.
2. else문이 결코 실행되지 않을 것 같다면 -> else내에 사용자 지정 프로그램 종료 함수 호출
3. if문은 2단계 이하로만 중첩
4. if문을 문단처럼 다룬다.
5. Bool값 검사는 간단하게

For문

1. 형식: for in문 형식만 가능

For 변수 in 반복 가능한 객체:

…반복 실행 구문…

1. 반복 가능한 객체

: 리스트, 딕셔너리, 집합, 튜플, byte, range

For와 range

1. 형식: range(시작값, 종료값, 스탭)
2. 리스트를 조각낸것과 같음.
3. 시작값과 종료값 바로 전 단계 숫자까지 포함
4. 값 확인은 다른 순서가 있는 컬랙션으로 변환

For와 enumerate

1. 몇번째 반복문인지 확인이 필요할 때 사용
2. 인덱스 번호와 원소를 tuple로 반환

While문

1. 형식

While 조건:

…반복 실행 구문…

1. 조건이 참인 동안 반복 수행

: 조건 제어는 while문 내에서 처리

1. for문과 while문 선택 기준

: 반복 횟수가 정해지지 않을 때

: 반복횟수가 정해진 경우

While 문 모범 사례

1. for문으로 가능한지 검토
2. 조건이 언제 거짓이 되는지 확인
3. 의심시 순환블록 아래 검사 변수 출력
4. 무한루프 조심: ctrl+c 로 종료

Break와 continue

1. Break 문

: 특정 조건에 반복 문 중단시키고 빠져 나옴

1. Continue 문

: 반복문을 중단시키지 않고 다음 반복으로 넘어감

1. python 객체 지향 프로그래밍

모듈

: 함수, 변수,클래스 등의 python코드로 이뤄진 파일

: import를 통해 사용 -> 파일명만 사용

: . 연산자로 멤버 접근

클래스

1. 메서드와 데이터를 함꼐 다루기 위한 자료형
2. 소형 모듈을 만드는 청사진이나 정의
3. 클래스 내의 멤버는 . 연산자로 접근
4. 모듈과 클래스 비교

: 모듈은 한 프로그램에서 딱 한번 존재

: 클래스는 한 프로그램에서 여러 모양으로 존재 가능

1. 형식

: self – 관례, 호출된 객체 자신 전달

Class className:

Def \_init\_(self, param):

클래스 본문

객체

1. Instantiate

: 모듈의 임포트에 해당

: 결과는 객체

1. 인스턴스화 형식

: 변수 = className(object)

1. 생성자

: 객체 생성에 사용

: def \_new\_(cls)

: 클래스 자체를 받으며 할당

: 자동으로 실행되므로 생략

1. 초기자

: def \_init(self)

: 객체 내부에서 사용할 속성 초기화

객체 생성시 전달되는 값 검증 수행

Oop 용어 정리

1. Class 선언
2. Object
3. Def
4. Self
5. Inheritance
6. Composition
7. Attribute
8. Is a: ~이다
9. Has a : ~을 가지고 있다.

객체와 클래스의 관계

1. 객체 = 특성+행위+정체성
2. 클래스 = 데이터 구조 + 메서드

: 클래스로 구현한 객체 -> 인스턴스

: 객체를 만드는 과정 -> 인스턴스 화

상속

1. 부모 클래스의 자산을 자식 클래스가 물려 받음
2. 기본형식

: 자식 클래스 선언시 괄호로 부모 클래스 포함

: 부클래스 자산 기술 필요 없음

Class parentclass:

…내용…

Class childclass(parentclass):

…내용…

Is-a와 has-a

1. Is-a 관계

: 클래스 관계로 엮인 객체와 클래스

: 상속 관계

1. Has-a 관계

: 클래스와 객체가 서로 참조로 엮인 경우

: 합성 관계

추상 클래스

1. 추상 메서드를 하나 이상 가진 클래스
2. 자식 클래스에서 추상 메서드 구현 강제
3. Abc 모듈 import
4. 형식

Class abstractclass(metaclass=abcmeta):

@abcstractmethod

Def abstract\_method(self):

Pass

1. numpy와 데이터 구조의 이해

python을 쓰면 안돼는 경우

1. 성능
2. 멀티쓰레드 프로그래밍

Python data analysis

1. Numpy
2. Pandas
3. Matplotlib
4. Ipython
5. Scipy

Numpy

: 빠르고 효율적인 다차원 배열 객체

: 배열 원소를 다루거나 배열간의 수학 계산을 수행하는 함수

: 파일로부터 배열 기반의 데이터를 읽거나 쓰기

: 선형대수, 푸리에 변환 및 난수 생성

: 타 프로그래밍 언어, 포트란 코드와의 통합

1 AI란 무엇인가?

: 인간의 능력을 모방하여 만들어진 것

* 데이터와 과거 경험을 기반으로 의사 결정 ex) 갑작스러운 소나기가 올 때
* 비정상적인 이벤트 인식: 상황이 잘못되었음을 인지
* 시각적 입력 해석
* 서면 및 음성 언어 이해
* 대화 참여

AI의 주 활동

1. 머신러닝, 기계학습

: 과거의 데이터 및 통계를 기반으로 모델이나 함수를 만들고 그것을 기반으로 미래의 데이터가 들어왔을 때 예측함.

1. 이상 탐지, 변칙검색

: 정상적인 상황이 아닌 경우 잘못된 상황을 인식하여 조치를 가능하게 하는 시스템

1. Computer vision

: 카메라, 이미지 또는 비디오의 시각적 입력을 해석하는 애플리케이션

1. 자연어 처리

: 서면 또는 음성언어를 해석할 수 있는 애플리케이션

1. 대화형 AI

: 인간의 대화에 참여할 수 있는 AI에이전트 혹은 봇

마이크로소프트 인공지능 기능

: 클라우드, 플랫폼 기반 AI

* 확장성 있음
* 데이터 저장 공간, 클라우드 제공 컴퓨팅 혹은 서버 리소스, 서비스 필요
* Azure machine learning

: 기계 학습 모델의 학습, 배포 및 관리를 위한 플랫폼

: 알고리즘을 만들어 모델링 하고 새로운 데이터를 이용해서 검증 및 예측함.

: 가장 큰 특징은 no coding 으로 drag & drop 기능 사용

* Cognitive services

: 개발자가 AI솔루션을 구축하는 데 사용할 수 있는 서비스 제품 군

* Azure bot service

: 봇 개발 및 관리를 위한 클라우드 기반 플랫폼

AI가 가지고 있는 도전과 위험성은 무엇일까?

1. 편견

: 과거의 데이터를 그대로 사용한다면 결과에 영향을 줄 수 있음.

1. 오류
2. 데이터 노출 위험성
3. 솔루션이 누구에게나 통하지 않을 수 있음.
4. 사용자는 복잡한 시스템을 신뢰해야 함.
5. AI가 주도한 결정의 책임.

책임성 있는 AI가 되기 위한 원칙

1. 공정성
2. 신뢰 및 안정성
3. 보안 및 개인정보
4. 포괄적
5. 투명성
6. 책임감

2 머신러닝

머신러닝의 종류

1. 회귀
2. 분류
3. 클러스터링

머신러닝이란?

: 데이터에서 관련성을 찾아 예측하는 모델을 만드는 것

Regression 회귀분석

: 과거의 데이터를 기반으로 모델링을 통해 미래를 예측하는 것

* 방법

1. 데이터를 training/validation data로 쪼갬.
2. 이후 training data를 validation과 비교하여 얼마나 정확한지 확인
3. 그래프를 작성하여 모든 데이터를 포함하는 선을 찾기
4. 구한 함수식으로 미래의 데이터를 예측함

Classification 분류

: training data를 통해 함수 식 작성, validation을 통해 확인

: positive(1) or negative(0)로 나뉘어짐.

: true – 실제 값과 예측 값이 같은 경우

: false – 실제 값과 예측 값이 서로 다른 경우

: true positive – 결과 값과 예측 값이 서로 1인 경우, false positive – 결과 값은 0인데 예측 값은 1인 경우

: false negative – 결과 값은 1인데 예측 값은 0인 경우, true negative – 결과 값은 0인데 예측 값은 0인 경우

Clustering 군집 분석

: 컴퓨터에게 비지도 학습 시킴.

1. 각 분포의 중심점을 찍음
2. 중심점으로부터 가까운 데이터를 하나로 묶음

Azure machine learning

: 클라우드 기반 머신러닝 플랫폼

4 자연어 처리

자연어 처리 종류

1. 텍스트 분석 및 엔티티 인식: 텍스트에서 명사(엔티티)만 뽑아내는 기능
2. 감정 분석: 글을 분석하여 글이 긍정적인지 부정적인지 파악
3. 음성인식 및 합성
4. 기계번역: 외국어를 모국어로 번역하여 음성으로 변환
5. 의미 체계적 언어 모델링

5 챗 봇

대화형 AI란?

: AI에이전트와 인간 간에 대화를 가능하게 하는 솔루션

: 일반적으로 봇으로 알려져 있음

: 여러 채널에 참여 가능(이메일, 음성, SNS 등)

봇의 책임

1. 자신의 가능 유무를 따져야 함.
2. 사용자와 확실히 통신 가능
3. 필요시 봇에서 인간으로의 전달이 원할해야함.
4. 문화적 규범 존중
5. 신뢰 가능해야 함.
6. 사용자 개인 정보 보호
7. 데이터 보안 처리
8. 접근성 표준을 충족 해야 함.
9. 복의 행동에 대한 책임을 져야 함.

The 붐 maker service

: 질문 및 답변 상의 기술 자료

: 봇을 포함한 클라이언트 앱 기술

Azure bot service

: 봇 개발 및 관리를 위한 클라우드 기반 플랫폼

: 여러 채널과 연결 가능