파이썬

상명대학교 융합공과대학
지능•데이터융합학부
휴먼지능정보공학전공
dkim@smu.ac.kr

강의개요

- 강의소개 및 프로그래밍 개념
  - 프로그래밍과 컴퓨팅사고력 소개
  - 프로그래밍 맛보기
- 변수, 자료형, 연산, 함수
  - 코딩과 기초실습
- 조건문, 연산자
  - 코딩과 기초실습
- 반복문
  - 코딩과 기초실습
- 함수, 매개변수
  - 코딩과 기초실습
- 중간고사

강의개요

- 자료형, 리스트
  - 코딩과 기초실습
- 자료형, 튜플
  - 코딩과 기초실습
- 자료형, 딕션너리
  - 코딩과 기초실습
- 실습예제
  - 코딩과 기초실습
- 파일읽고 쓰기
  - 코딩과 기초실습
- 객체지향 프로그래밍
  - 코딩과 기초실습
- 기말고사

프로그래밍 문법

- 변수: 데이터를 저장하는 공간
  - 구별되게 고유한 이름을 붙임

- 자료형(타입) : 데이터(자료)를 표현하는 방법(형태(타입)가 있음)
  - 수치형(정수,실수,복소수), 문자열(따옴표), 리스트, 튜플, 딕셔너리, 집합

- 연산자: 프로그램의 산술식이나 연산식을 표현하고 처리하기 위해 제공되는 기호
  - 대입연산, 산술연산, 복합대입연산

- 자료형(타입)변환: 데이터(자료)를 표현하는 방법을 변환
  - 문자열연산, 정수형 변환int()함수)

프로그래밍 문법

- 리스트(배열): 자료구조 형태중 하나로 순서가 있는 수정 가능한 데이터의 집합
  - 접근, 수정, 삭제, 추가가 가능
  - [] 대괄호로 작성되어지며, 내부 원소는 ,로 구분
  - 리스트 이름 = [요소1, 요소2, 요소3,…]
- 문자열: 자료구조 형태 중 하나로 순서가 있는 문자 데이터의 나열
  - a="상명대"
    - >>> print(a) → 상명대
- 튜플: 자료구조 형태중 하나로 순서가 있는 수정 불가능한 데이터의 집합
  - 접근, 삭제 가능, 추가 불가능
  - ( )괄호로 작성되어지며, 내부 원소는 ,로 구분
  - 튜플 이름 = (요소1, 요소2, 요소3,…)
- 딕셔너리: 자료구조 형태 중 하나로 순서가 없는 쌍으로 수정 가능한 데이터의 집합
  - 접근, 추가, 삭제 가능
  - 중괄호({ })로 묶여 있으며 키와 값의 쌍으로 이루어지고 내부 원소는, 로 구분
  - 딕셔너리 이름= {키1:값1, 키2:값2, 키3:값3,…}
  - a = {1:"가", 2:"나", 3:"다", 4:"라", 5:"마"}
    - >>> print(a) → {1: '가', 2: '나', 3: '다', 4: '라', 5: '마'}
- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합

집합

- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합
  - 접근, 연산 가능, 추가, 제거 가능, 중복허용 안됨
  - set 키워드를 이용, set()함수
  - set() 괄호 안에 리스트, 문자열 입력 가능
  - a = set(“상명대학교상명”)
    - >>> print(a) → {'교', '명', '학', '상', '대'}
  - 인덱스로 접근하려면 리스트 또는 튜플로 변환 (순서가 있는 데이터 집합으로 변환)

```
a = set(“상명대학교상명”)
print(a)
b = list(a)
print(b)
c = sorted(b)
print(c)
→
{'교', '명', '학', '상', '대'}
['교', '명', '학', '상', '대']
['교', '대', '명', '상', '학']
```

집합

- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합
  - 접근, 연산 가능, 추가, 제거 가능, 중복허용 안됨
  - set 키워드를 이용, set()함수
  - set() 괄호 안에 리스트, 문자열 입력 가능
  - a=set(1,2,3,4,5,1,2)
    - print(a) → TypeError: set expected at most 1 arguments, got 7
  - a=set([1,2,3,4,5,1,2])
    - print(a) → {1, 2, 3, 4, 5}
    - print(len(a)) → 5

집합

- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합
  - 집합 연산 → 합집합(OR,union), 교집합(AND,intersection), 차집합(-,difference)
  - a = set([1,2,3,4,5,1,2]) ## a={1,2,3,4,5,1,2}
  - b= set([2,4,6,8,10,2,4]) ## b={2,4,6,8,10,2,4}
    - print("집합a:",a) → 집합a: {1, 2, 3, 4, 5}
    - print("집합b:",b) → 집합b: {2, 4, 6, 8, 10}
    - print("합집합:",a|b) → 합집합: {1, 2, 3, 4, 5, 6, 8, 10}
    - print("합집합:",set.union(a,b)) → 합집합: {1, 2, 3, 4, 5, 6, 8, 10}
    - print("합집합:",a.union(b)) → 합집합: {1, 2, 3, 4, 5, 6, 8, 10}
    - print("교집합:",a&b) → 교집합: {2, 4}
    - print("교집합:",set.intersection(a,b)) →교집합: {2, 4}
    - print("교집합:",a.intersection(b)) →교집합: {2, 4}
    - print("차집합:",a-b) → 차집합: {1, 3, 5}
    - print("차집합:",set.difference(a,b)) →차집합: {1, 3, 5}
    - print("차집합:",a.difference(b)) →차집합: {1, 3, 5}

집합

- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합
  - 집합 연산 → 부분집합(subset), 진부분집합(proper subset)
  - 부분집합: 어떤 집합의 원소 중 일부만을 포함하는 집합
  - 진부분집합: 원소의 크기가 더 작은 부분집합
  - a =set([1,2,3,4,5,1,2]) ## a={1,2,3,4,5,1,2}
  - b=set([1,2,3,4,5]), ##b={1,2,3,4,5}
  - c=set([2,4]) ## c={(2,4}
    - print("부분집합",a.issubset(b)) → 부분집합 True
    - print("부분집합",b.issubset (a)) → 부분집합 True
    - print("부분집합",c.issubset(a)) → 부분집합 Ture
    - print("진부분집합",a.issuperset(b)) → 진부분집합 True
    - print("진부분집합",b.issuperset(a)) → 진부분집합 True
    - print("진부분집합",c.issuperset(a)) → 진부분집합 False

집합

- 집합: 자료구조 형태 중 순서가 없는 수정 가능한 데이터의 집합
  - 집합 연산 → 집합의 크기(cardinality)
  - a =set([1,2,3,4,5,1,2]) ## a={1,2,3,4,5,1,2}
  - b=set([1,2,3,4,5]), ##b={1,2,3,4,5}
  - c=set([2,4]) ## c={(2,4}
    - print(len(a)) → 5
    - print(len(b)) → 5
    - print(len(c)) → 2

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력: 키보드로 입력, 모니터로 출력
    - 표준 입력: input() → 키보드, 문자열
    - 표준 출력: print() → 모니터, 문자열

  - 파일 입출력: 파일로 입력(읽음), 파일로 출력(씀)
    - 파일 입력(읽음)
      - read()
      - readline()
      - readlines()
    - 파일 출력(씀)
      - write()
      - writelines()

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 표준 입력: input() → 표준 출력: print()
    - 표준 입력: input() → 파일 출력: write(), writellness()
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - 파일 입력: read(), readline(), readlines() → 파일 출력: write(), writelines()

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 파일열기
    - open() 함수에서 파일명을 지정, 읽기인지 쓰기인지를 지정
    - open() 함수의 마지막 매개변수를 모드
      - 변수이름 = open("파일이름", "r") → 파일읽기(입력)
      - 변수이름 = open("파일이름", "w") → 파일쓰기(출력)

      - 파일열기모드
        - r: 읽기모드 기본값
        - w: 쓰기모드 기본에 파일이 있으면 덮어씀
        - r+: 읽기/쓰기 겸용 모드
        - a: 쓰기모드, 기존에 파일이 있으면 이어서 씀 (Append)
        - t: 택스트모드, 텍스트 파일을 처리, 기본값
        - b: 바이너리 모드, 바이너리 파일(이진파일) 처리

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 파일처리
    - 파일에 데이터 쓰거나 파일로부터 데이터를 읽어올 수 있는 상태
  - 파일닫기
    - 변수이름.close()

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
      - 메모장에 텍스트 입력:

        컴퓨터 프로그램 어떤 문제를 해결하기 위해 컴퓨터에게 주어지는 처리 방법과 순서를 기술한 일련의 명령문의 집합

        컴퓨터 프로그래밍 컴퓨터가 이해할 수 있는 규칙에 따라 프로그램 수행절차를 프로그래밍 언어로 작성하는 것

        프로그래밍 알고리즘은 프로그래밍 언어를 사용하여 어떠한 문제를 해결하기 위한 명령어 모임
      - 메모장 텍스트 저장: 인코딩 UTF-8로 저장
      - 메모장 파일 이름: data.txt

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - read() → 일정 길이 만큼 텍스트를 읽음

```
inFile = None
inString =“”
inFile = open(“C:/test/data.txt”, “r”, encoding=“utf-8”)
inString = inFile.read(2) # 값-1
print(inString, end=“”)
inFile.close()
→
컴
```

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - readline() → 텍스트를 한 줄씩 읽음

```
inFile = None
inString =“”
inFile = open(“C:/test/data.txt”, “r”, encoding=“utf-8”)
inString = inFile.readline()
print(inString, end=“”)
inString = inFile.readline()
print(inString, end=“”)
inString = inFile.readline()
print(inString, end=“”)
inString = inFile.readline()
print(inString, end=“”)
inFile.close()
```

→

컴퓨터 프로그램 어떤 문제를 해결하기 위해 컴퓨터에게 주어지는 처리 방법과 순서를 기술한 일련의 명령문 집합
컴퓨터 프로그래밍 컴퓨터가 이해할 수 있는 규칙에 따라 프로그램 수행절차를 프로그래밍 언어로 작성하는 것
프로그래밍 알고리즘은 프로그래밍 언어를 사용하여 어떠한 문제를 해결하기 위한 명령어 모임

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - readline() → 텍스트를 한 줄씩 읽음

```
inFile = None
inString =“”
inFile = open(“C:/test/data.txt”, “r”, encoding=“utf-8”)
while True :
        inString = inFile.readline()
        if inString==“”:
                break;
        print(inString, end=“”)
inFile.close()
```

→

컴퓨터 프로그램 어떤 문제를 해결하기 위해 컴퓨터에게 주어지는 처리 방법과 순서를 기술한 일련의 명령문 집합

컴퓨터 프로그래밍 컴퓨터가 이해할 수 있는 규칙에 따라 프로그램 수행절차를 프로그래밍 언어로 작성하는 것

프로그래밍 알고리즘은 프로그래밍 언어를 사용하여 어떠한 문제를 해결하기 위한 명령어 모임

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - readlines() → 텍스트 모든 줄을 한번에 읽음

```
inFile = None
inList = []
inString = “”
inFile = open(“C:/test/data.txt”, “r”, encoding=“utf-8”)
inList = inFile.readlines()
for inString in inList:
        print(inString, end=“”)
inFile.close()
```

→

컴퓨터 프로그램 어떤 문제를 해결하기 위해 컴퓨터에게 주어지는 처리 방법과 순서를 기술한 일련의 명령문 집합
컴퓨터 프로그래밍 컴퓨터가 이해할 수 있는 규칙에 따라 프로그램 수행절차를 프로그래밍 언어로 작성하는 것
프로그래밍 알고리즘은 프로그래밍 언어를 사용하여 어떠한 문제를 해결하기 위한 명령어 모임

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 파일 입력: read(), readline(), readlines() → 표준 출력: print()
    - readlines() → 텍스트 모든 줄을 한번에 읽음

```
import os
def file_read_test(inFile):
    inList, inString = [], ""
    if os.path.exists(inFile) :
        inFile = open(inFile, "r", encoding="utf-8")
        inList = inFile.readlines()
        for inString in inList:
            print(inString, end="")
        inFile.close()
    else :
        print(“%s 파일이 없습니다”%(inFile)

def test():
    inFile = input("파일 이름을 입력하세요")
    file_read_test(inFile)

test()
 →
파일 이름을 입력하세요C:/test/data.txt
```

파일입출력

- 파일입출력: 파일 입력과 출력 과정에 필요한 함수
  - 표준 입출력과 파일 입출력
    - 표준 입력: input() → 파일 출력: write(), writellness()

```
outFile = None
outString = ""
outFile = open(" C:/test/data2.txt", "w", encoding="utf-8")
while True :
    outString = input("메모를 입력하세요:")
    if outString !="" :
        outFile.writelines(outString +"\n")
    else :
        break
outFile.close()
```

→

```
메모를 입력하세요:안녕하세요
메모를 입력하세요:
```

→

data2.txt 생성여부 확인

프로그래밍 설계

프로그래밍 상세명세

- 1) 메뉴판을 출력하고 사용자로부터 투입 금액을 받고 투입금액이 최소금액(500원)보다 적으면 투입 금액 부족 문자열을 출력합니다
- 2) 투입금액이 최소금액보다 크면 사용자로부터 음료메뉴 선택과 수량을 입력 받습니다
- 3) 선택한 음료와 수량으로 결제 총액을 계산하고 총액이 투입금액보다 크거나 같으면 결제 금액이 부족하다는 문자열을 출력합니다
- 4) 결제 총액이 투입금액보다 적으면 거스름돈을 계산하고
- 5) 사용자에게 선택한 음료의 종류와 수량 결제금액과 잔액을 출력합니다

프로그래밍 구현

프로그래밍 상세명세

- 1) 메뉴판을 출력하고 사용자로부터 투입 금액을 받고 투입금액이 최소금액(500원)보다 적으면 투입 금액 부족 문자열을 출력합니다
- 2) 투입금액이 최소금액보다 크면 사용자로부터 음료메뉴 선택과 수량을 입력 받습니다
- 3) 선택한 음료와 수량으로 결제 총액을 계산하고 총액이 투입금액보다 크거나 같으면 결제 금액이 부족하다는 문자열을 출력합니다
- 4) 결제 총액이 투입금액보다 적으면 거스름돈을 계산하고
- 5) 사용자에게 선택한 음료의 종류와 수량 결제금액과 잔액을 출력합니다

```
==============================================
==          상명대 파이썬 자판기             ==
==1.캔커피 800원  2.생수 500원  3.소다 900원==
==============================================
동전을 입력하세요1000
음료를 선택하세요2
수량을 입력하세요1

선택하신 음료는 생수이고 수량 1개입니다.
결제금액은 500원 잔액은 500원 입니다.
이용해주셔서 감사합니다
>>>
```

프로그래밍 구현

프로그래밍 상세명세

- 1) 메뉴판을 출력하고 사용자로부터 투입 금액을 받고 투입금액이 최소금액(500원)보다 적으면 투입 금액 부족 문자열을 출력합니다
- 2) 투입금액이 최소금액보다 크면 사용자로부터 음료메뉴 선택과 수량을 입력 받습니다
- 3) 선택한 음료와 수량으로 결제 총액을 계산하고 총액이 투입금액보다 크거나 같으면 결제 금액이 부족하다는 문자열을 출력합니다
- 4) 결제 총액이 투입금액보다 적으면 거스름돈을 계산하고
- 5) 사용자에게 선택한 음료의 종류와 수량 결제금액과 잔액을 출력합니다

```
==============================================
==          상명대 파이썬 자판기             ==
==1.캔커피 800원  2.생수 500원  3.소다 900원==
==============================================
동전을 입력하세요1000
음료를 선택하세요2
수량을 입력하세요3
결제 금액이 부족합니다
>>>
```

프로그래밍 설계

프로그래밍 명세

- 프로그램 메뉴를 표시한다
- 1번 메뉴는 사용자에게 자판기 음료 메뉴를 알려준다
- 2번 메뉴는 사용자에게 자판기 음료 재고 현황을 알려준다
- 3번 메뉴는 사용자에게 음료를 판매한다.
    - 사용자로부터 투입금액을 입력 받는다
        - 투입금액이 500원 이하면 판매할 수 없다
    - 사용자로부터 음료 이름을 입력 받는다
        - 자판기에서 판매하는 음료 아니면 판매할 수 없다
    - 사용자로부터 음료 수량을 입력 받는다
        - 자판기 재고량보다 많이 입력하면 판매 할 수 없다
    - 사용자에게 선택한 음료 재고를 알려준다
        - 자판기 재고량이 소진되면 판매할 수 없다
    - 사용자에게 총액을 알려준다
    - 사용자에게 거스름돈을 알려준다
        - 자판기 거스름돈(잔고)가 부족하면 판매할 수 없다
    - 사용자에게 잔고를 알려준다.
    - 영수증을 저장한다.
- 4번 영수증을 출력한다
    - 영수증 출력(판매시간, 음료이름, 음료수량, 투입금액, 총액, 잔액(거스름돈), 승인번호(무작위), 감사합니다
- 5번 메뉴는 프로그램을 종료한다.

프로그래밍 설계

```
===== 상명 파이썬 자판기 프로그램 (Ver1.0) =====

        1.메뉴 보기
        2.재고 현황
        3.음료 판매
        4.영수증 보기
        5.프로그램 종료하기

--------------------------------------------------
현재 잔고는 5000 원
==================================================
```

```
메뉴를 선택하세요1
--------------------------------------------------
*** 상명 파이썬 자판기 메뉴  ***
음료:캔커피      가격:800원
음료:소다        가격:600원
음료:생수        가격:500원
음료:오렌지      가격:900원
--------------------------------------------------
```

```
메뉴를 선택하세요2
--------------------------------------------------
*** 음료 재고 현황 ***
음료:캔커피      재고:5개
음료:소다        재고:5개
음료:생수        재고:5개
음료:오렌지      재고:5개
--------------------------------------------------
```

프로그래밍 설계

```
메뉴를 선택하세요3
투입 금액을 입력하세요1000
사용자가 1000원 투입했습니다
음료를 선택하세요생수
생수가 메뉴에 있습니다
수량을 선택하세요1
생수음료 재고가 4개 남았습니다
------------------------------------------------
*** 음료 재고 현황 ***
음료:캔커피      재고:5개
음료:소다        재고:5개
음료:생수        재고:4개
음료:오렌지      재고:5개
------------------------------------------------
사용자가 생수를 1개 선택했습니다
총액은 500원 입니다
거스름돈은 500원 입니다
현재 잔고는 5500원 입니다
이용해주셔서 감사합니다!!!
```

```
메뉴를 선택하세요4
--------영수증-------
판매시간:2020/05/24 21:21:46
음료이름 생수
음료수량:1
투입금액:1000
총액:500
잔액(거스름돈):500
승인번호:7032
이용해주셔서 감사합니다 !!
```

맺음말