과제 1

장유선

2023.08.02

1. 문제 정의

입력받은 데이터를 읽어 아래의 값을 확인하여 아래의 형태로 ‘result.txt’로 출력

* 30000 이하의 값 개수
* 최소값
* 최대값
* 평균

**<Input Example>**

|  |
| --- |
| 17425  10383  62468  44527  ... |

**<Output Example>**

|  |
| --- |
| 30000 이하의 값: 243개  최소값: 12  최대값: 99234  평균: 3453.234643 |

2. Python Code Hard Copy

|  |
| --- |
| f = open('input.txt', 'r', encoding='utf-8')  lines = f.read().splitlines()  data = []  count = 0  sum = 0  for line in lines:      num = line.strip('\ufeff')      num = int(num)      data.append(num)      sum += num      if num <= 30000:          count += 1      minimum = min(data)      maximum = max(data)      avg = sum / len(data)      with open('result.txt', 'w', encoding='utf-8') as f:          f.write(f"30000 이하의 값: {count}개\n")          f.write(f"최소값: {minimum}\n")          f.write(f"최대값: {maximum}\n")          f.write(f"평균: {avg:.6f}\n")  with open('result.txt', 'r', encoding='utf-8') as f:      lines = f.read().splitlines()  for line in lines:      print(line) |

3. Code 분석

3-1. ‘input.txt’ 파일 읽기

f = open('input.txt', 'r', encoding='utf-8')

lines = f.read().splitlines()

<함수 설명>

1. **open()**

open(‘input.txt’, ‘r’, encoding=’utf-8’

input.txt 파일을 읽기 모두(‘r’)로 열고, 인코딩은 ‘utf-8’을 사용하여 파일을 읽는다.

* encoding = ‘utf-8’

유니코드 문자열은 단순히 문자 셋의 규칙이기에 유니코드 문자열은 인코딩(encoding) 없이 그대로 파일에 적거나 다른 시스템으로 전송할 수 없다 파일에 적거나 다른 시스템으로 전송하려면 바이트(byte) 문자열로 변환해야 한다. 이렇게 유니코드 문자열을 바이트 문자열로 바꾸는 것을 ‘인코딩’이라고 한다. UTF-8은 표준 유니코드 문자를 모두 표현할 수 있다. 한글이 포함된 소스는 UTF-8로 인코딩을 한다.

1. **read()**

파일 객체 f를 이용하여 ‘input.txt’ 파일의 전체 내용을 읽는다.

주의) 파일 내용을 문자열로 반환한다.

1. **splitliess()**

read()로 읽은 문자열을 줄 단위로 분리한다. 각 줄이 리스트의 항목으로 저장된다.

3-2. 변수 선언

data = []

count = 0

sum = 0

<함수 설명>

1. **data**

파일에서 읽은 숫자들을 저장하는 리스트이다. 각 숫자들을 int() 함수를 이용하여 정수로 변환한 후, 이를 data 리스트에 추가한다.

1. **count**

파일에서 읽은 숫자들 중 30000 이하의 값을 가진 숫자들의 개수를 저장하는 변수이다.

1. **sum**

파일에서 읽은 숫자들의 합을 저장하는 변수이다. 평균을 계산할 때 이 값을 이용한다.

3-3. 총합, 최소, 최대, 평균 계산

for line in lines:

    num = line.strip('\ufeff')

    num = int(num)

    data.append(num)

    sum += num

    if num <= 30000:

        count += 1

    minimum = min(data)

    maximum = max(data)

    avg = sum / len(data)

<함수 설명>

1. **num = line.strip('\ufeff')**

문자열 맨 앞에 BOM(BYTE ORDER MARK) 문자가 있다면 이를 제거하기 위해 strip(‘/ufeff’)메서드를 사용한다. strip() 함수는 문자열의 양쪽 끝에 있는 특정 문자들을 제거하는 파이썬의 문자열 메서드이다. 이 함수는 문자열을 수정하는 것이 아니라, 제거된 새로운 문자열을 반환한다. BOM 문자는 유니코드 문자열의 가장 처음에 위치하는 바이트 순서 표시로 사용되기 때문에 제거해야한다.

1. **data.append(num)**

정수로 변환된 num 값을 data 리스트에 추가한다.

1. **sum += num**

sum 변수에 num 값을 누적 더한다. 파일에서 읽은 숫자들의 총 합을 계산한다.

1. **If num <= 30000:**

**count += 1**

num 값이 30000 이하인 경우에만 count 변수를 증가시킨다. 30000 이하의 값을 가진 숫자들의 개수를 셀 수 있다.

1. **minimum = min(data)**

**maximum = max(data)**

**avg = sum / len(data)**

data 리스트에 저장된 숫자들을 통해 최소값, 최대값, 평균을 계산한다. min(), max() 내장 함수를 이용하여 최소값과 최대값을 반환한다. 총합에 data의 길이 즉 숫자의 개수를 나누어서 평균을 계산한다.

3-4. ‘result.txt’로 저장

 with open('result.txt', 'w', encoding='utf-8') as f:

        f.write(f"30000 이하의 값: {count}개\n")

        f.write(f"최소값: {minimum}\n")

        f.write(f"최대값: {maximum}\n")

        f.write(f"평균: {avg:.6f}\n")

<함수 설명>

1. **with open('result.txt', 'w', encoding='utf-8') as f:**

3-1의 설명과 같다.

1. **write()**

파일 객체를 이용하여 파일에 데이터를 쓰는 메서드이다.

3-5. 출력

with open('result.txt', 'r', encoding='utf-8') as f:

    lines = f.read().splitlines()

for line in lines:

    print(line)

1. **for line in lines:**

**print(line)**

파일의 각 줄을 순회하면서, 각 줄의 내용을 출력한다. 파일의 내용이 줄 단위로 출력된다.

4. 출력 결과

30000 이하의 값: 1390개

최소값: 22

최대값: 64982

평균: 32586.385000

5. 출력 화면

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명