재귀와 반복 - 검색

Recursion & Iteration - Searching

실습 1 : 마지막 문자열 하나만 찾기

파일이름 filename과 찾을 문자열 key을 받아서 파일에서 문자열이 마지막으로 나타나는 위치번호를 "result.txt" 파일에 쓰는 프로시저 find_last를 만들어보자.

```
1  def find_last(filename,key):
.
.
.
.
```

즉, 텍스트 파일이름이 "article.txt"이고 찾으려는 문자열이 "컴퓨터"이면, find_last("article.txt","컴퓨터")를 호출하고, "article.txt"에서 "컴퓨터" 가 마지막으로 나타나는 위치번호를 "result.txt" 파일에 쓰고, 없으면 not found를 쓴다.

귀띔 : key를 찾는 작업을 끝까지 반복해서 한다. 최근 찾은 위치 하나는 항상 기억하고 있다가 끝나면 기억해 둔 위치를 쓴다.

프로시저를 완성하고 다음 호출을 실행하여 확인해보자.

```
find_last("article.txt","컴퓨터")   # 컴퓨터의 위치번호는 6355
find_last('article.txt','데스크탑') # 데스크탑의 위치번호는 6360
find_last("article.txt","한양대")   # 한양대는 not found
```

실습 2 : 모두 찾기

파일이름 filename과 찾을 문자열 key를 받아서 파일에서 문자열이 나타나는 위치번호를 모두 "result.txt" 파일에 쓰는 프로시저 find_all을 만들어보자.

```
1  def find_all(filename,key):
.
.
.
.
```

즉, 텍스트 파일이름이 "article.txt"이고 찾으려는 문자열이 "컴퓨터"이면, find_all("article.txt","컴퓨터")를 호출하고, "article.txt"에서 "컴퓨터"를 모두 찾아 위치번호를 차례로 "result.txt" 파일에 쓰고, 없으면 not found를 쓴다.

프로시저를 완성하고 다음 호출을 실행하여 확인해보자.

```
find_all("article.txt","컴퓨터")   # 총 12번 등장
find_all("article.txt","데스크탑") # 총 1번 등장
find_all("article.txt","한양대")   # 없음
```

실습 3 : 모두 찾고, 찾은 개수 세기

파일이름 filename과 찾을 문자열 key을 받아서 파일에서 문자열이 나타나는 위치번호를 모두 "result.txt" 파일에 쓰고 마지막으로 찾은 횟수를 쓰는 프로시저 find_all_count을 만들어보자.

```
1  def find_all_count(filename,key):
.
.
.
.
```

즉, 텍스트 파일이름이 "article.txt"이고 찾으려는 문자열이 "컴퓨터"이면, find_all_count("article.txt","컴퓨터")를 호출하고, "article.txt"에서 "컴퓨터"가 나타나는 위치번호를 모두 "result.txt" 파일에 쓰고, 없으면 not found를 쓴 뒤, 총 나타난 횟수를 맨 뒤에 쓴다.

프로시저를 완성하고 다음 호출을 실행하여 확인해보자.

```
find_all_count("article.txt","컴퓨터")    # 총 12번 등장
find_all_count("article.txt","데스크탑")  # 총 1번 등장
find_all_count("article.txt","한양대")    # 총 0번 등장
```

실습 4 : 한 줄에 한문장씩 오게 입력 파일을 재편성 하기

텍스트 파일(테스트용으로 "article.txt" 사용)을 읽은 뒤, 한 줄에 한 문장이 오도록 파일을 재편성하여 "result.txt" 파일에 쓰는 one_sentence_per_line 프로시저를 다음 포맷에 맞추어 작성하시오.

```
1  def one_sentence_per_line(filename) :
2      infile = open(filename,"r")
3      outfile = open( “ result.txt ” ,"w")
4      text = infile.read()
5      count = 0
.      …
.      …
.      outfile.write("문장이 총 " + str(count) + "개\n")
.      outfile.close()
.      infile.close()
.      print("done")
```

각 문장 사이에 빈 줄을 한 줄 씩 띄어야 한다. 문장은 마침표(".") 또는 물음표("?")로 끝나므로 전에 공부한 partition 메소드를 사용하여 한 문장씩 분리해 낼 수 있다. 맨 아래 줄에 문장의 개수를 다음과 같이 표시한다.

문장이 총 162개

실습 5 : 특정 문자열이 들어있는 문장 모두 찾기

파일이름 filename과 문자열 key가 주어지면, key를 포함하고 있는 문장을 모두 찾고, 그 문장에 key가 몇 번 등장했는지 다음과 같이 "result.txt" 파일에 쓰는 find_all_sentence 프로시저를 작성하시오.

'컴퓨터'이(가) 3번 등장
(마이크로소프트의) 윈도즈는 맥 컴퓨터를 단지 베낀 것에 불과하기 때문에, 맥 컴퓨터가 그렇게 하지 않았다면 어떤 개인용 컴퓨터도 그런 아름다운 서체를 갖지 못했을 것입니다.

추가로 맨 아래 줄에 몇개의 문장에서 몇개의 key가 등장하는지 다음과 같이 표시한다.

'컴퓨터'이(가) 8개 문장에서 12번 등장

find_all_sentence 프로시저는 다음 형식을 맞추어 작성하시오.

```
1  def find_all_sentence(filename,key) :
2      infile = open(filename,"r")
3      outfile = open("result.txt","w")
4      text = infile.read()
.      …
.      …
.      outfile.close()
.      infile.close()
.      print("done")
```