CSED101. Programming & Problem solving
Fall 2021

Programming Assignment #3
(75 points)

허성환 (hursung1@postech.ac.kr)

■ **제출 마감일: 2021.11.30 23:59**

■ **개발 환경: Windows Visual Studio 2019**

■ **제출물**

- C 소스 코드 및 출력 파일 (mystring.h, mystring.c, assn3.c)
  - 프로그램의 소스 코드에 채점자의 이해를 돕기 위한 **주석**을 반드시 붙여주세요.
- 보고서 파일 (.docx, .hwp 또는 .pdf; assn3.docx, assn3.hwp 또는 assn3.pdf)
  - 보고서는 AssnReadMe.pdf를 참조하여 작성하시면 됩니다.
  - 보고서는 Problem2에 대해서만 작성해 주세요.
  - 명예 서약 (Honor code): 표지에 다음의 서약을 기입하여 제출해 주세요: "나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다." 보고서 표지에 명예 서약이 기입되어 있지 않은 과제는 제출되지 않은 것으로 처리됩니다.
  - 작성한 소스 코드와 보고서 파일은 PLMS를 통해 제출해 주세요.

■ **주의 사항**

- 컴파일이나 실행이 되지 않는 과제는 0점으로 채점됩니다.
- 제출 기한보다 하루 늦게 제출할 때 20%, 이틀 늦게 제출할 때 40% 감점됩니다. 제출 기한보다 사흘 이상 늦으면 제출 받지 않습니다 (0점 처리).
- 각 문제의 제한 조건과 요구 사항을 반드시 지켜 주시기 바랍니다.
- 모든 문제의 출력 형식은 아래의 예시들과 동일해야 하며, 같지 않을 시 감점됩니다.
- 부정행위에 관한 규정은 POSTECH 전자컴퓨터공학부 학부위원회의 "POSTECH 전자컴퓨터공학부 부정행위 정의"를 따릅니다 (PLMS의 본 과목 공지사항에 등록된 글 중, 제목이 [document about cheating]인 글에 첨부되어 있는 disciplinary.pdf를 참조하세요).
- 이번 과제는 추가 기능 구현과 관련된 추가 점수가 따로 없습니다.

[들어가기 전]

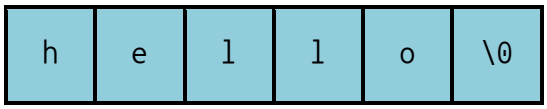
1. 문자열(string)

- 연속된 문자들로 C 언어에서 문자열 앞 뒤에 "를 이용한다.
- char 형의 1차원 배열을 이용하여 문자열을 저장한다.
- 배열에 문자열을 저장할 때는 끝에 NULL 문자 ('\0')를 넣어서 표시한다.

2. 선언

- **char str[] = "hello";**
  위와 같이 선언과 동시에 초기화를 하게 되면, 자동으로 문자열의 끝에 NULL 문자가 추가된다.

  str



3. 입력과 출력

```
char str[100];
printf("Enter the filename: ");
scanf("%s", str);
printf("%s", str);
```

<실행 예시> (아래 예시의 밑줄은 사용자 입력에 해당)

```
Enter the filename: train.txt
train.txt
```

■ Problem1: String functions (3점)

(문제)

C의 <string.h> 라이브러리에서 제공하는 문자열 처리 함수 중 일부를 직접 구현한다.

(설명)

제공된 assn3_p1.zip파일 내에 mystring.h와 mystring.c가 존재한다. mystring.h는 아래와 같은 문자열 처리 함수의 선언을 포함하고 있다. (그대로 사용하고 변경하지 말 것)

```
#pragma once

int mystrlen(char *str);
int mystrcmp(char *str1, char *str2);
char *mystrcpy(char *toStr, char *fromStr);
char *mystrlower(char *str);
```

위 함수의 구현부를 포함한 mystring.c를 작성한다. 각 함수의 정의는 아래와 같다.

(1) int mystrlen(char *str)

NULL문자를 제외한 문자열의 길이를 return 한다. 빈 문자열(" \0 ")의 경우 0을 return 한다.

Ex)

```
printf("%d\n", mystrlen("csed101")); //결과: 7
```

(2) int mystrcmp(char *str1, char *str2)

문자열 str1과 str2의 대소관계를 비교한다. 비교 기준은 ASCII이다. 각 문자열의 첫 문자부터 비교하며, 문자가 같으면 다음 문자를 다시 비교한다. return 할 값은 다음의 기준에 따라 정한다.

- 비교 중 str1의 문자가 작은 경우 -1, 클 경우 1을 return
- 비교하는 두 문자열이 완전히 같은 경우(길이가 같고 모든 문자가 같음) 0을 return
- 비교 중 한 문자열이 먼저 끝에 도달하면 그 문자열이 더 작다고 판단

Ex)

```
string = "csed101"
printf("%d\n", mystrcmp(string, "csed101"));  //결과: 0
printf("%d\n", mystrcmp(string, "csed103"));  //결과: -1
printf("%d\n", mystrcmp(string, "csed"));     //결과: 1
printf("%d\n", mystrcmp(string, "csed1010")); //결과: -1
printf("%d\n", mystrcmp(string, "Csed101"));  //결과: 1
```

(3) char *mystrcpy(char *toStr, char *fromStr)

문자열 fromStr을 문자열 toStr에 복사한 후, 문자열 toStr의 주소를 return 한다.

Ex)

```
char str[100];
printf("%s\n", mystrcpy(str, "Hello world!")); // 결과: Hello world!
printf("%s\n", mystrcpy(str, "CSED101"));      //결과: CSED101
```

(4) char *mystrlower(char *str)

문자열 str 내의 모든 영어 알파벳을 소문자로 바꾼 후, 문자열 str의 주소를 return 한다.

```
char str[100] = "Hello World! 123";
printf("%s\n", mystrlower(str)); //결과: hello world! 123
```

■ Problem2: Naïve Bayes Classifier를 활용한 스팸문자 분류 (72점)

[목적]

- 포인터와 동적 할당 사용법을 익힌다.
- 파일 입출력의 사용법을 익힌다.
- String의 사용법을 익힌다.

[배경지식]

1. 들어가기

우리가 메일 클라이언트(ex. 구글 지메일, 네이버 메일 등)를 사용할 때, 자동으로 스팸 메일로 분류하여 차단해주는 경우가 있다. 자연언어처리의 큰 task 중 하나인 text classification은 위 예시와 같이 어떤 문장 혹은 글을 특정한 범주로 분류하는 작업을 수행한다. 이를 위한 수많은 방법이 존재하는데(tf-idf, support vector machine, …), 이들 중 한 가지 간단한 방법인 **Naïve Bayes Classification**은 **Bayes' Theorem**을 기초로 하여 text classification을 수행한다.

2. Bayes' Theorem

두 확률 변수의 결합 확률에 대해, 다음과 같은 식이 성립한다.

$$p(x,y) = p(x|y)p(y) = p(y|x)p(x)$$

여기서, $p(x|y)p(y) = p(y|x)p(x)$ 만을 사용하여, 다음과 같이 표현할 수 있다.

$$p(y|x) = p(x|y)p(y)/p(x)$$

위에서 언급한 스팸 필터링의 예시를 들어 위 식을 살펴보자. $p(y|x)$는 **어떤 메일(x)이 스팸인지 아닌지(y)를 결정하는 확률**이다. 관측 가능한 정보(ex. 메일 데이터 셋)를 통해 우변의 식의 확률 값들을 구하여, 좌변의 값을 계산하는 것이 Bayes' Theorem의 목적이다.

우변을 살펴보면, 먼저 $p(y)$는 **특정 범주의 등장 확률**이다. 이는 현재 데이터 셋 내의 스팸 메일의 등장 확률을 사용한다. $p(x|y)$는 **특정 범주에 대한 그 메일의 등장 확률**을 나타낸다. $p(x)$는 해당 메일의 등장 확률을 나타내지만, 보통은 계산 편의를 위해 이 값을 직접 구하지 않고 $p(x|y)p(y)$와의 비례 관계만을 이용한다.

3. Naïve Bayes Classification

x는 여러 단어들$(w_1, w_2, \dots, w_n)$로 이루어진 문서(혹은 문장)이므로, $p(x|y)$는 이 단어들의 **조건부 결합 확률**로 볼 수 있다.

$$p(x|y) = p(w_1, w_2, \dots, w_n|y)$$

Naïve Bayes Classification은 여기서 **각 단어의 등장 사건을 조건부 독립으로 가정**한다. 독립의 정의에 따라 식을 다음과 같이 변형할 수 있다.

$$p(x|y) = p(w_1, w_2, \dots, w_n|y) = \prod_i p(w_i|y)$$

즉, 우리가 x를 분류하고자 할 때, 다음과 같은 식으로 분류할 수 있다.

$$y = argmax_y p(y) \prod_i p(w_i|y)$$

여기서 $argmax_y$는 식의 값을 최대화하는 y를 의미한다.

4. 예시

다음과 같은 training data가 있다고 하자.

I will buy a new phone	**Ham**
buy our product, change your old phone	**Spam**
pay attention to our new product	**Spam**
let's discuss about our homework	**Ham**
I think it is new one	**Ham**

어떤 문장 x = "buy our new phone"을 Ham인지 Spam인지 분류하고자 한다. 즉 $p(Ham|x)$와 $p(Spam|x)$를 구하고자 한다. Naïve Bayes Classification 방법에 따라, 먼저 다음의 확률 값들을 구한다.

- $p(y)$
  - $p(Ham) = 3/5,\ p(Spam) = 2/5$
  - ※ $p(Spam)$: 주어진 학습 데이터에서 스팸의 비율을 말함. 즉, 5개의 데이터 중에 2개가 스팸이므로 2/5 로 계산 함

- $p(w_i|y)$
  - $p(buy\,|\,Ham) = 1/3,\quad p(buy\,|\,Spam) = 1/2$
  - $p(our\,|\,Ham) = 1/3,\quad p(our\,|\,Spam) = 1$
  - $p(new\,|\,Ham) = 2/3,\quad p(new\,|\,Spam) = 1/2$
  - $p(phone\,|\,Ham) = 1/3,\quad p(phone\,|\,Spam) = 1/2$
  - ※ 'new'라는 단어가 학습 데이터 5개 중에 2개는 정상 메일(Ham)에, 1개는 스팸에 있으므로 $p(new\,|\,Ham) = 2/3,\ p(new\,|\,Spam) = 1/2$ 로 계산 함

이를 통해 $p(y)\prod_i p(w_i|y)$값을 구하면 다음과 같다.

- $p(Ham)\prod_i p(w_i|Ham) = 2/135$
- $p(Spam)\prod_i p(w_i|Spam) = 1/20$

아래의 값이 더 크므로, x는 Spam**으로 분류**할 수 있다.

[주의사항]

1. 파일 이름은 "assn3.c"로 저장하고, 보고서는 problem2에 대해서만 작성한다.
2. 표준 헤더 파일 <string.h>를 include하여 사용할 수 있다.
3. 전역 변수, goto 문, 구조체 및 연결리스트(linked list)를 사용할 수 없다.
4. 모든 기능을 main 함수에 모든 기능을 구현한 경우 감점 처리한다. 기능적으로 독립됐거나 반복적으로 사용되는 기능은 사용자 함수를 정의해서 구현한다.
5. 문제 설명에서 메모리 동적 할당을 요구하는 부분에서 배열 사용시 감점된다.
6. 명시한 에러 처리 외에는 고려하지 않아도 된다.
7. 문제의 출력 형식은 "=-=-" 출력을 제외하고 아래의 예시들과 동일하게 작성해 주세요.

[구현 기능 설명]

0. 프로그램 시작

실행 시 다음과 같이 메인 화면을 출력 후, 사용자 입력을 위해 대기한다.

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: _
```

이때, 사용자는 3개의 명령어(train, test, exit) 중 하나를 입력하여 해당 기능을 수행한다. 3개의 명령어 외 입력은 적절한 에러메시지 출력 후, **Sleep()** 함수를 사용하여 1초간 대기한 후, **system("cls")**를 사용하여 화면을 지우고 메인 화면을 다시 보여준다.

(예시의 밑줄은 사용자 입력에 해당)

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: print
Error: Invalid input
```

명령어 입력 시, 대소문자를 구분하지 않고 동일한 명령어의 기능을 수행하도록 한다. 예를 들면 test, Test, TEST 등은 동일한 기능을 수행한다.

1. Training

"train"을 입력하면 다음과 같이 파일 이름을 입력 받아, 해당 파일을 읽도록 한다.

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: train
File name: train.txt
```

파일 이름을 저장할 배열은 아래와 같이 선언하고 사용한다. 참고로, 실제 채점 시에는 20자 이내의 파일 이름을 입력하여 테스트 할 예정이다. 파일 이름에는 공백이 없다고 가정한다.

```
#define MAX_FILE_NAME 30
char filename[MAX_FILE_NAME];
```

파일은 본 과제와 함께 제공된 train dataset 포맷을 가진다. 아래 예시는 제공한 train.txt 파일의 내용으로, Train dataset 파일은 모든 line이 아래와 같이 **[label]\t[text]** 형태로 구성되어 있다(**\t**는 **tab 문자**를 의미).

train dataset의 예시)

```
ham     I will buy a new phone
spam    buy our product, change your old phone
spam    pay attention to our new product
ham     let's discuss about our homework
ham     I think it is new one
```

만약 **입력 받은 파일이 존재하지 않는 경우**, 다음과 같이 에러 메시지를 출력한다. 현재 상태를 1초간 유지 후, 화면을 지우고 메인 화면을 다시 보여준다.

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: train
Error: File does not exist
```

[label]은 ham/spam 둘 중 하나이며, [text]의 길이는 1000자를 넘지 않는다. 확률 값을 구하기 위해 단어의 등장 빈도 수를 구해야 하는데, 다음의 처리 과정이 필요하다.

- Problem 1에서 구현한 함수를 사용하여 **[text] 내 모든 알파벳을 소문자화** 한다.
  ※ mystring.h를 include하여 사용해도 되고, 해당 코드만 가져와서 사용해도 된다.
- [text]에 포함된 특수 문자, 즉 **알파벳을 제외한 모든 문자는 공백으로 치환**한다.

위 과정이 끝나면, 각 [label]에 대한 각 단어의 등장 빈도 수를 구한다. (단어-등장 횟수) 쌍을 저장하기 위해 다음과 같은 형태로 배열을 구성해야 한다. 이때 단어(char **words)와 등장 횟수(int *freq)를 저장할 배열은 고정 배열을 사용하지 않고, 각각 동적 할당 받아 생성한다. 즉 **필요할 때 마다 메모리를 동적 할당/재할당**한다. 처음 할당 시에는 배열의 크기를 5만큼 할당하고, 부족한 메모리를 재할당 하고자 할 때, 현재 할당된 크기의 2배를 재할당한다(ex. 현재 크기: 10 → 재할당 후 크기: 20).

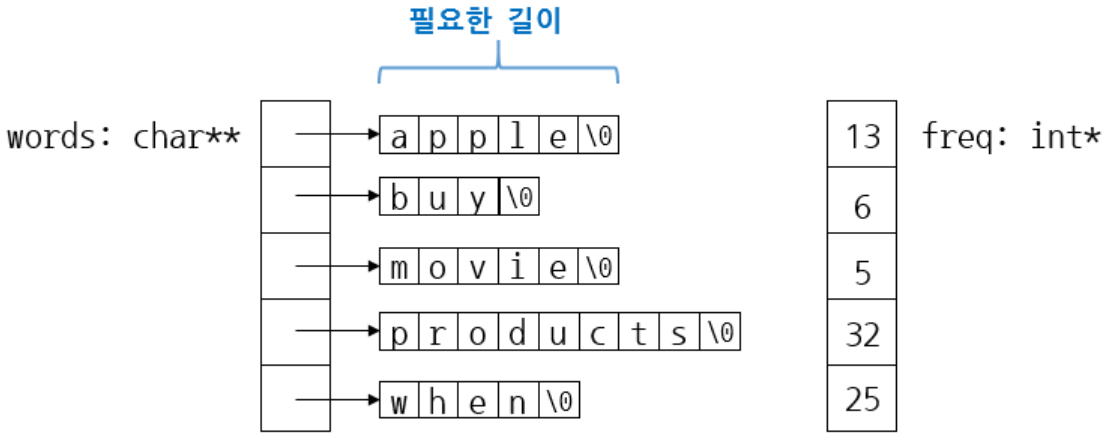


그림 1 데이터 구성 방법의 한 예

단어의 경우, 각 단어마다 필요한 길이(단어의 문자열 길이 + 1)만큼 동적 할당 받아 해당 단어를 저장하도록 한다.

아래는 문자열 길이만큼 메모리를 동적 할당한 예제이다. 입력 받을 문자열의 길이는 미리 알 수 없기 때문에 충분한 크기의 배열(str)을 선언하여 그 곳에 저장하고, 입력 받은 문자열을 저장할 길이만큼만 메모리를 동적 할당하여 단어를 저장한다.

```
#define MAX_WORD_LEN 30
…
char str[MAX_WORD_LEN]; // 문자열을 입력 받을 배열
char *p;                // 동적 메모리 할당을 받을 포인터
scanf("%s", str);       // "hello"를 입력 받은 경우, "hello\0"를 저장하는 공간을
                        // 제외한 나머지 메모리가 낭비
p = (char *)malloc(sizeof(char) * (strlen(str)+1)); // str에 저장된 문자열의 크기와
                                                    // 동일한 공간 할당
strcpy(p, str);         // p가 가리키는 공간에 문자열 복사 기능 수행
…
free(p);                // p에 대한 사용이 모두 끝나면, 메모리 할당 해제
```

단어 등장 빈도 수를 구하는 데 있어 다음의 주의 사항을 따른다.

- **단어의 길이가 1인 경우** 빈도 수를 세지 않는다.
- **단어의 길이가 20을 초과하는 경우** 빈도 수를 세지 않는다.
- 위 두 가지 경우를 제외한 나머지 경우에만 빈도 수를 구한다.
- 하나의 메시지 내에 단어가 여러 번 등장하더라도, **등장 여부만을 고려**하여 한 번만 센다.

학습이 종료되면, 위 과정을 통해 구한 통계를 **단어 기준으로, 알파벳순으로 정렬** 후 아래의 예시처럼 화면에 출력하고 **파일로 저장**한다. Training을 위해 할당한 메모리는 **free() 함수를 통해 정상적으로 반환**한다.

출력 파일 이름은 '*stats.txt*'로 하며, 화면 또는 파일에 출력할 때 다음의 주의 사항을 따른다.

- 결과 출력의 첫번째 줄은 train dataset에서 학습한 정상 메일(Ham)과 스팸 메일의 개수를 아래 예시처럼 기록한다.
- 두 번째 줄부터 출력 형식은 **[단어],[Ham의 경우 등장 빈도 수],[Spam의 경우 등장 빈도수]** 로 한다.
  만약 어느 쪽에 해당 단어가 없는 경우라면 **0으로 출력한다**. 예를 들어, "can"이라는 단어가 Ham의 경우 10번 등장했으나, Spam의 경우 등장하지 않았다면, "can,10,0"으로 출력한다.

아래 예시는 데이터 셋으로 제공한 'train.txt' 파일을 이용하여 학습 후, 그 통계를 화면에 출력한 예시이다.

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: train
File name: train.txt

After training...
Ham:3, Spam:2      <-------- 통계 결과 출력 시작 부분
about,1,0
attention,0,1
buy,1,1
change,0,1
discuss,1,0
homework,1,0
is,1,0
it,1,0
let,1,0
new,2,1
old,0,1
one,1,0
our,1,2
pay,0,1
phone,1,1
product,0,2
think,1,0
to,0,1
will,1,0
your,0,1
```

통계를 출력한 후, 사용자의 입력을 기다린다. [Enter]키를 입력하면 화면을 지우고, 메인 화면을 다시 보여준다.

아래 예시는 데이터 셋으로 제공한 'train.txt' 파일을 이용하여 학습 후, 그 통계를 파일('stats.txt')로 출력한 예시이다.

```
Ham:3, Spam:2
about,1,0
attention,0,1
buy,1,1
change,0,1
discuss,1,0
homework,1,0
is,1,0
it,1,0
let,1,0
new,2,1
old,0,1
one,1,0
our,1,2
pay,0,1
phone,1,1
product,0,2
think,1,0
to,0,1
will,1,0
your,0,1
```

2. Test

"test"를 입력하면 다음과 같이 문장의 입력을 기다린다.

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: test
Enter a message:
```

문장을 입력하면, 프로그램은 training 단계에서 구한 통계('stats.txt') 파일을 기반으로 해당 문장이 Spam인지 아닌지를 판별한다. 단, 문장의 길이는 1000자를 넘지 않는다.

다음의 주의사항을 따른다.

- Training 단계에서 처리했던 것과 마찬가지로, **문장 내 특수 문자를 모두 제거**하며, 모든 알파벳은 **소문자화** 한다.

- 문장 내 단어가 통계에 **포함되지 않는 경우**, 해당 단어에 대한 확률 값은 **계산에 포함하지 않는다**.
- 통계에 포함되었으나 개수가 0인 경우 확률 값이 0이 되므로, 이 경우 확률 값을 **0.1/(Spam인 경우의 수) 또는 0.1/(Ham인 경우의 수)**로 한다.
- **test는 training을 거치지 않고 바로 수행**될 수 있다. 이 경우 test에 사용할 파일의 이름은 "**stats.txt**"이어야 하고, **training 단계에서 생성한 파일과 동일한 포맷**을 가지고 있어야 한다.

파일을 읽는 것은 사용자로부터 문장을 입력 받기 전에 수행하도록 한다. 파일에서 정보를 불러올 때, 고정 크기의 배열에 저장하는 것이 아니라 training 단계와 마찬가지로 **포인터를 선언하여 메모리를 동적으로 할당/재할당**하여야 한다. (이 때, 필요하면 10진 문자열을 int 형 정수 값으로 변환하는 atoi()함수 사용 가능, 사용시 stdlib.h를 포함시킬 것)

만약 입력 받은 문장이

- Spam인 것으로 판별되면: "This message is SPAM"을 출력한다.
- Ham인 것으로 판별되면: "This message is HAM"을 출력한다.

Spam 판별 메시지와 함께 통계 정보를 출력한다. test를 위해 할당한 메모리는 **free() 함수를 통해 정상적으로 반환**한다.

아래의 예시는 제공한 train.txt의 예시를 사용하여 구한 stats.txt를 기반으로 구한 통계 정보이다(계산 과정은 4~5페이지의 [배경지식]과 예시를 참고). 계산 값은 소수점 셋째 자리까지 출력한다. (ex. 0.1 → 0.100, 0.3126 → 0.313)

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: test
Enter a message: buy our new phone

P(Ham) = 0.600, P(Spam) = 0.400
P(buy | Ham) = 0.333, P(buy | Spam) = 0.500
P(our | Ham) = 0.333, P(our | Spam) = 1.000
P(new | Ham) = 0.667, P(new | Spam) = 0.500
P(phone | Ham) = 0.333, P(phone | Spam) = 0.500

P( Ham | 'buy our new phone'): 0.015
P(Spam | 'buy our new phone'): 0.050

This message is SPAM
```

위 화면에서 사용자의 입력을 기다린다. [Enter]키를 입력하면 화면을 지운 후 다시 메인 화

면을 보여준다.

만약 **training을 하지 않은 상태**에서, test 명령을 입력하였으나 **"stats.txt" 파일이 없는** 경우, 다음과 같이 에러 메시지를 출력한다. train에서와 마찬가지로 1초 간 대기한 후, 화면을 지운 뒤 다시 메인 화면을 보여준다.

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naive Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: test
Error: File does not exist
```

3. 프로그램 종료

"exit"를 입력하면 프로그램을 종료한다. 이 때 동적으로 할당된 메모리는 **free() 함수를 통해 정상적으로 반환**하여야 한다.

※ 다른 training data 를 사용한 예

Training data)

```
ham     are you free tomorrow? why don't we go ice rink
spam    U R entitled to Update to the latest color mobiles with camera for Free!
ham     i see :) see you later
spam    You have won a 1 week FREE membership in our $100,000 Prize!
spam    The New Jersey Devils and the Detroit Red Wings play Ice Hockey. Correct or
Incorrect? Reply END SPTV
ham     did you buy notebooks? If not i will buy them
spam    you are awarded with a $1500 Bonus Prize, call 09066364589
```

stats.txt)

```
Ham: 3, Spam: 4
and,0,1
are,1,1
awarded,0,1
bonus,0,1
buy,1,0
call,0,1
camera,0,1
color,0,1
correct,0,1
detroit,0,1
devils,0,1
did,1,0
don,1,0
end,0,1
entitled,0,1
for,0,1
free,1,2
go,1,0
have,0,1
hockey,0,1
ice,1,1
if,1,0
in,0,1
incorrect,0,1
jersey,0,1
later,1,0
latest,0,1
membership,0,1
mobiles,0,1
new,0,1
not,1,0
```

```
notebooks,1,0
or,0,1
our,0,1
play,0,1
prize,0,2
red,0,1
reply,0,1
rink,1,0
see,1,0
sptv,0,1
the,0,2
them,1,0
to,0,1
tomorrow,1,0
update,0,1
we,1,0
week,0,1
why,1,0
will,1,0
wings,0,1
with,0,2
won,0,1
you,3,2
```

Test 결과)

```
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
CSED 101
Assignment 3

Naïve Bayesian Classifier for Spam Filtering
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
Command: test
Enter a message: you got a $1000 lottery! Call us :)

P(Ham) = 0.429, P(Spam) = 0.571

P(you | Ham) = 1.000, P(you | Spam) = 0.500
P(call | Ham) = 0.033, P(call | Spam) = 0.250

P( Ham | 'you got a $1000 lottery! Call us :)'): 0.014
P(Spam | 'you got a $1000 lottery! Call us :)'): 0.071

This message is SPAM
```

[참고]

과제 수행 시 다음 내용을 참고하세요.

1. 공백을 포함한 문자열 입력

   공백이 있는 문자열을 입력 받을 때 fgets() 함수를 사용한다.

   char *fgets(char *str, int num, FILE *stream)

   : stream에서 문자열을 최대 num-1개만큼 받아서 str이 가리키는 메모리에 저장한다. \n이 포함된 경우 그 문자열의 길이가 num보다 작더라도 더 읽지 않는다. stdio.h에 포함.

   Ex)

```
char str[10];
fgets(str, 10, stdin); // stdin은 키보드로부터 입력 받는 것을 의미
printf("%s\n", str); // 입력 받은 문자열을 출력

FILE *fp = fopen("file.txt", "r");
fgets(str, 10, fp); // 파일에서부터 한 라인을 읽음
printf("%s\n", str); // 파일로부터 읽은 문자열을 출력
```

2. 문자열 분할

   특정 문자를 기준으로 문자열을 분할하고자 할 때, strtok() 함수를 사용한다.

   char *strtok(char *str, const char *delim)

   : 문자열 str을 delim에 포함된 문자들로 분리(tokenize)한다. 예를 들어 "Hello/world!"라는 문자열에 대해 "/"를 기준으로 분리하면 "Hello"와 "world!"로 나뉘어진다. 이 때 기준이 되는 문자("/")를 delimiter, 나누어진 문자들을 token이라고 한다. string.h에 포함.

   Ex)

```
char str[100];
char *token1, *token2, *token3;
strcpy(str, "Tokenization/Test/String");
token1 = strtok(str, "/"); // token1 = "Tokenization"
token2 = strtok(NULL, "/"); // token2 = "Test"
token3 = strtok(NULL, "/"); // token3 = "String"
```