빅데이터 HW2 (문장생성확률 계산 – 음절단위)

컴퓨터공학부

20163084 권보경

*** 알고리즘**

1) 전체 음절에 대한 count 변수를 두고 횟수를 셀 때마다 count++를 해주어 전체 음절의 출현 횟수를 구함

2) (각 음절의 출현 횟수 / 전체 음절 횟수) 를 계산하여 각 음절의 출현 확률을 계산

3) 각 음절의 출현 확률을 모두 곱하여 문장 발생 확률(음절단위)를 계산

*** input 텍스트**

확확확률률계산

*** UTF8 실행 결과**

```
gwonbogyeong-ui-MacBook-Pro:빅 데 이 터  bokyeong$ ./a.out
utf8
freq[계 ] = 1
probablity[계 ] = 0.142857
freq[률 ] = 2
probablity[률 ] = 0.285714
freq[산 ] = 1
probablity[산 ] = 0.142857
freq[확 ] = 3
probablity[확 ] = 0.428571
count : 7.000000
probability of sentence : 0.000131
```

***KS완성형 실행 결과**

```
gwonbogyeong-ui-MacBook-Pro:鐵  ꡓ 댄 ꡓ bokyeong$ ./a.out
ks
freq[계 ] = 1
propability[계 ] = 0.142857
freq[률 ] = 2
propability[률 ] = 0.285714
freq[산 ] = 1
propability[산 ] = 0.142857
freq[확 ] = 3
propability[확 ] = 0.428571
count : 7.000000
probability of sentence : 0.000131
```

*** 전체 코드**

```
#include <stdio.h>

int countKS[256][256]={0,};
int countUTF[65536];
float count = 0.0;
long double prob = 1.0;

int KS_or_UTF8(FILE* f)
{
	char c1, c2;
	c1 = fgetc(f);

	if((c1 & 0xf0) == 0xe0) //utf8
	{
		printf("utf8₩n");
		return 1;
	}
	else //ks
	{
		printf("ks₩n");
		return 0;
	}
}

void freqKS(FILE* f)
{
	int c1, c2;

	f = fopen("test.txt", "r");

	while(!feof(f))
	{
		c1 = fgetc(f);
		if(feof(f)) break;

		if ((c1 & 0x80) == 0)
		{	// MSB == 0
			countKS[0][c1]++;
```

```
                        // printf("freq[%c%c] = %d₩n", 0, c1, countKS[0][c1]);
                        count++;

                } else {
                        c2 = fgetc(f);
                        countKS[c1][c2]++;          // DBCS
                        // printf("freq[%d%d] = %d₩n", c1, c2,countKS[c1][c2]);
                        count++;

                }
        }
}

void output_KSC5601()
{
    int i, j;

    for (i=0xA1; i <= 0xFE; i++) {
        for (j=0xA1; j <= 0xFE; j++) {
                if (countKS[i][j] >= 1)
                {
                        printf("freq[%c%c] = %d₩n", i, j, countKS[i][j]);
                        printf("propability[%c%c] = %f₩n", i, j, countKS[i][j]/count);
                        for(int k=0; k<countKS[i][j]; k++)
                        {
                                prob *= (countKS[i][j]/count);
                        }
                }
        }
    }
}

void freqUTF(FILE* f)
{
        int c1, c2, c3;

        f = fopen("test.txt", "r");

        while(!feof(f))
```

```
        {
            c1 = fgetc(f);
                if(feof(f)) break;

            if ((c1 & 0x80) == 0) { // MSB == 0
                countUTF[c1]++;
                                    // printf("freq[] = %d₩n", countUTF[c1]);
                                    count++;
            } else {
                c2 = fgetc(f);
                c3 = fgetc(f);
                int i = (c1 & 0x0f) << 12 | (c2 & 0x3f) << 6 | (c3 & 0x3f);
                countUTF[i]++;      // DBCS
                 // printf("freq[] = %d₩n", countUTF[i]);
                 count++;

            }
        }
}

void output_UTF8()
{
    int i,j;
    char utf8[4] = {0};

    for (i=0xAC00, j=0; j<11172; i++, j++) {
        utf8[0] = 0xE0; utf8[0] |= ((i>>12) &0x000F);
        utf8[1] = 0x80; utf8[1] |= ((i>>6) &0x003F);
        utf8[2] = 0x80; utf8[2] |= (i&0x003F);

        if (countUTF[i] >= 1)
        {
                printf("freq[%c%c%c] = %d₩n", utf8[0],utf8[1],utf8[2], countUTF[i]);
                printf("probablity[%c%c%c] = %f₩n", utf8[0], utf8[1], utf8[2], countUTF[i]/count);
                for(int k=0; k<countUTF[i]; k++){
                        prob *= (countUTF[i]/count);
                }
        }
    }
```

```
}

int main()
{
	char c1, c2, c3;

	FILE *f = fopen("test.txt", "r");

	if(KS_or_UTF8(f) == 1)
	{
		freqUTF(f);
		output_UTF8();
	}
	else
	{
		freqKS(f);
		output_KSC5601();
	}

	printf("count : %f ₩n", count);
	printf("probability of sentence : %Lf ₩n", prob);

	return 0;
}
```