CSE3081 (2반): 알고리즘 설계와 분석
(HW 3)

담당 교수: 임 인 성

2021년 11월 16일 (v1.0)

마감: 11월 28일 일요일 오후 8시 정각
제출물, 제출 방법, LATE 처리 방법 등: 조교가 사이버 캠퍼스에 공지할 예정임.

목표: 수업 시간에 다룬 Dynamic Programming과 Greedy Method에 대한 이해를 높인다.

1. **[Problem 1]** 다음과 같은 문제를 생각하자.

   "NURSESRUN"처럼 앞에서부터 읽건 뒤에서부터 읽건 동일한 문자열을 회문(palindrome)이라고 한다. 어떤 문자열의 부분 문자열은 원래의 문자열에서 일부 문자만 뽑아 순서를 유지하면서 나열한 문자열으로서, "ABDGI"는 "ABCDEFGHIJ"의 부분 문자열이다. 한 문자열의 부분 문자열 중 회문이면서 길이가 가장 긴 것을 *longest palindromic subsequence (LPS)*라 한다(예를 들어, "BBABCBCAB"의 LPS는 "BABCBAB"임). 이제 주어진 문자열 $X = x_0x_1 \cdots x_{m-1}$의 LPS를 찾는 문제를 생각하자.

   이 문제를 해결해주는 다음과 같은 함수를 작성하라.

   ```
   void LPS(const int m, const char *X, int *LPS_length,
            char **LPS_string);
   ```

   (a) 이 함수는 i) 포인터 변수 `X`가 가리키는 길이가 `m`인 입력 문자열의 LPS를 구한 후, ii) 그 LPS에 대한 메모리를 할당 받아 저장한 후, iii) 그 포인터를 `**LPS_string`에 저장하고, iv) 그 LPS의 길이를 `*LPS_length`에 저장하여 리턴해주는 방식으로 작동해야 한다. (반드시 양식을 지킬 것)

   (b) 여러분의 프로그램은 이름이 `config_LPS.txt`인 텍스트 파일(ASCII 파일)에서 필요한 정보를 읽어 들여 그에 맞게 작동해야 한다. 다음의 예제 파일에서,

   ```
   3
   input1.bin
   output1.bin
   input21.bin
   output21.bin
   myinput17.bin
   myoutput17.bin
   ```

   첫 줄에는 몇 개의 데이터에 대해 문제를 해결해야하는지 정수값(`3`)이 오고, 다음 그 회수 만큼 두 파일의 이름이 반복적으로 나열이 된다. 각 쌍에 대해 첫 번째 파일(예를 들어, `input1.bin`)은 입력 문자열이 저장되어 있는 파일의 이름이며, 그에 대응되는 파일(`output1.bin`)은 여러분의 프로그램이 찾은 결과를 저장할 출력 파일에 해당한다.

   (c) 입력 문자열에 대한 입력 화일에는 이진(binary) 형식으로 데이터가 저장되어 있는데, 첫 4 바이트에는 X의 길이 m이 `int` 타입으로 저장되어 있고, 다음 m 바이트에는 X가 `char` 타입으로 저장이 되어 있음.

(d) 출력 파일도 역시 이진 (binary) 형식으로 저장이 되는데, 첫 4 바이트에는 여러분이 찾은 LPS의 길이 `LPS_length`가 `int` 타입으로 저장되고, 다음 연달아 `LPS_length` 바이트에는 여러분이 찾은 LPS가 `char` 타입으로 저장되어야 한다.

(e) 조교는 여러분의 원시 코드에 문제가 있는지를 확인한 후, 자신의 테스트 데이터를 사용하여 기계적으로 답을 확인할 예정이니 입출력 형식을 정확히 지킬 것.

2. **[Problem 2]** 임의의 번호를 가지는 카드 묶음을 고려하자. 이 문제에서는 동일한 번호를 가지는 카드가 여러 장 있을 수 있는데, 카드를 섞는 전형적인 방식은 카드 묶음을 둘로 나누어 한 손에 한 묶음씩 잡고 엄지와 검지에 힘을 주어 카드 허리를 휘게 한 다음, 양쪽의 카드를 조금씩 풀어 주면 '섞인 카드 묶음'을 얻게 된다. 이렇게 카드를 섞으면 잘 섞인 것 같지만, 사실 처음 양손에 가지고 있던 카드 묶음에서 카드의 순서는 섞인 카드 묶음에서도 그대로 유지된다. 예를 들어, 왼손에 `2, 4, 6, 8`의 순서로 카드를 가지고 있고, 오른손에 `1, 3, 5, 7`의 카드를 가지고 있다면, 위에서 설명한 방식으로 카드를 섞어서 얻을 수 있는 카드의 순서의 몇 가지 예를 들면 다음과 같다.

```
1, 3, 2, 4, 6, 5, 7, 8
1, 2, 3, 4, 5, 6, 7, 8
1, 3, 5, 7, 2, 4, 6, 8
```

그렇지만 아래와 같은 순서는 절대 얻을 수 없는데, 그 이유는 처음 것은 `3`과 `5`의 순서가 섞기 전과 섞은 후에 다르고, 두 번째 것은 `1`과 `3`의 순서가 다르기 때문이다.

```
1, 2, 4, 5, 6, 8, 3, 7
3, 1, 2, 4, 6, 5, 7, 8
```

이제 왼손에 있는 묶음의 카드 순서, 오른손 묶음의 카드 순서, 그리고 섞은 후 카드 묶음의 카드 순서가 입력될 때, dynamic programming 기법을 사용하여 양손의 카드 묶음을 위에서 설명한 것과 같은 방식으로 섞어서 만들 수 있는지 아닌지를 판별하는 프로그램을 작성하라 (이 문제에서는 카드 위치가 서로 다르지만, 번호가 동일한 카드가 제한 없이 입력될 수 있음). 여러분의 프로그램은 다음과 같은 입출력 요구사항을 만족해야 한다.

입력 형식
입력 데이터는 텍스트 파일에 저장되어 있다. 첫 째줄에는 왼손에 들고 있는 카드의 개수에 이어서 카드 순서가 입력되고, 두 번째 줄에는 오른손에 들고 있는 카드 묶음, 그리고 셋 째줄과 넷 째줄에는 한 줄에 하나씩 섞인 카드 묶음에 대한 정보가 같은 형식으로 입력된다. 왼손이나 오른손에 들고 있는 카드 개수는 1 이상 1,000장 이하이고, 카드 번호는 1부터 1,000 이하인 정수이다.

출력 형식
계산 결과는 역시 텍스트 파일에 저장되어야 한다. 한 줄에 왼손과 오른손에 들고 있는 카드 묶음을 위에서 정한 방식으로 섞어서 셋째 줄의 카드 순서를 얻을 수 있으면 1 아니면 0을 출력하고, 이어서 넷째 줄의 카드 순서를 얻을 수 있으면 1 아니면 0을 출력한다. 출력은 00, 01, 10, 11 중의 하나이다.

입출력 예 1

입력 (input.txt)
```
4 2 4 6 8
4 1 3 5 7
4 1 3 2 4 6 5 7 8
8 1 2 4 5 6 8 3 7
```

출력 (output.txt)
```
10
```

입출력 예 2

입력 (input.txt)
```
3 5 5 2
4 2 1 2 7
4 3 2 5 2 5 1 2 7
3 5 5 2 1 2 5 2 7
```
출력 (output.txt)
```
01
```

명령어 파일 (중요)
여러분의 프로그램은 이름이 (정확히) `commands_3_2.txt`인 텍스트 파일에서 한 줄에 한 개씩 기술되어 있는 입력 파일의 이름을 읽어들여 이름이 (정확히) `outputs_3_2.txt`인 텍스트 파일에 그 결과를 한 줄에 한 파일 결과씩 누적하여 출력해야 한다. 예를 들어, 다음과 같은 명령어 파일에 대하여,

```
input3.txt
input5.txt
input9.txt
```

결과는 다음과 같이 저장되어야 한다. (입출력 형식을 정확히 지킬 것)

```
00
10
01
```

3. **[Problem 3]** 다음의 문제를 greedy algorithm인 Huffman coding 방법을 사용하여 해결하여 보자.

   "Given a file, find a variable-length binary code for the characters in the file, which represents the file in the least number of bits."

   (a) 여러분이 작성하는 Huffman encoder의 입력 데이터는 자신의 프로그램 원시 파일과 동일한 디렉토리에 저장되어 있는 이름이 `P3_input_ASCII.txt`인 파일에 ASCII 형식의 텍스트로 저장되어 있다. Huffman encoder는 이 입력 데이터에 대한 처리를 수행한 후, 그 결과를 다음과 같이 두 파일에 나누어 입력 파일과 동일한 디렉토리에 저장해야 한다.

   i. **[출력 파일 1]** 입력 파일의 텍스트에 나타나는 문자들을 ASCII 코드 값을 기준으로 정렬한 후, 이름이 `P3_output_codewords.txt`인 파일에 다음과 같이 각 줄에 각 문자에 대한 정보를 순서대로 출력하라. 각 줄에는 (문자, codeword, 빈도 백분율) 정보를 정확히 아래와 같은 형식으로 출력해야 한다. 아래의 예에서 '8.51'은 문자 'a'가 이 파일에 8.51%의 빈도로 나타남을 의미하며, 백분율은 소수점 이하 두 자리까지 출력할 것.

   ```
   a 00 8.51
   b 110 4.57
   c 1110 3.63
   e 01 8.44
   ```
   ⋮

   이때 문자는 그림 1에 있는 모든 문자를 의미하며 이 파일에 문자 출력 시 그림 1 테이블의 `Chr (Char)` 컬럼의 문자를 사용하라.

   ii. **[출력 파일 2]** 이름이 `P3_output_encoded.bin`인 파일에 위의 codeword들을 사용하여 encoding한 결과를 binary 형식으로 출력하라. 이때 첫 네 바이트에는 입력 데이터를 최소 개수의 비트로 압축한 후의 비트 수, 그리고 다음 4 바이트에는 encoding된 binary 데이터가 차지하는 바이트 수를 저장한 후, 다음 그 바이트 수만큼의 공간에 encoding된 비트 정보를 패킹하여 저장하라. (앞의 두 수는 각각 `unsigned int` 형태로 저장하고, 마지막 바이트에 빈 공간이 존재한다면 비트 '0'으로 채울 것)

   (b) 조교는 채점 시 다음과 같은 방식으로 여러분의 프로그램의 정확성을 확인할 예정임.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Figure 1: ASCII Code Chart

- 주어진 입력 파일 `P3_input_ASCII.txt`의 압축에 필요한 최소 비트 수가 맞는지 확인함.
- 파일 `P3_output_codewords.txt`의 내용을 바탕으로 decoder 프로그램을 작성한 후, 결과 파일 `P3_output_encoded.bin`의 내용을 decoding한 후 그 내용이 `P3_input_ASCII.txt`의 내용과 '정확히 일치하는지 기계적으로 확인'함.

(c) Decoder 프로그램은 제출할 필요는 없으나, 자신이 구축한 Huffman 트리를 어떻게 하면 효과적으로 저장하고 또한 이를 바탕으로 어떻게 decoder 프로그램을 작성할 지 연습하여 볼 것.

[제출 방법]

1. 숙제 제출 내용 및 방식은 조교가 사이버 캠퍼스에 공지한 내용이 다음의 내용에 우선한다.

   (a) 프로그램 원시 코드 (자신이 사용한 입력 데이터는 제출하지 말것)

   - 위의 세 문제를 이름이 각각 **Problem_1**, **Problem_2**, 그리고 **Problem_3**인 디렉터리에 자신이 구현한 코드를 저장한 후 zip으로 묶어 제출하라. (Visual Studio의 `.vs` 디렉터리는 제거)
     주의: 조교는 이 코드들에 대하여 기계적으로 copy-check를 진행한 후 (역시) 기계적으로 채점을 할 예정이므로, 가급적 다른 학생의 코드와 비슷한 일이 발생하지 않도록 제출 코드에 자신만의 특생이 들어나도록 최선을 다할 것.

   (b) 보고서

   - 본인의 결과를 `HW3_S20197777.{hwp,docx,pptx,txt}`와 같은 이름의 보고서에 위에서 기술한 내용을 포함하여 간결하게 작성하라. 특히 부분 점수라도 받기 위해서는 자신이 구현한 항목과 구현하지 못한 항목에 대하여 정확히 구별하여 기술하라.

2. 숙제 제출 기간 동안 본 숙제와 관련하여 중요한 공지 사항을 사이버 캠퍼스에 올릴 수 있으니 항상 수업 게시판을 확인하기 바람.

3. 제출 화일에서 바이러스 발견 시 **본인 점수** x **(-1)**이고, 다른 사람의 숙제를 복사할 경우 **관련된 사람 모두에 대하여 만점** x **(-10)**임.