System Programming
Midterm Exam
Spring 2012

Part 1. Output Matching.

In this section, we have a file, named file, which has the following contents:
在這個部分，我們有一個叫做 "file" 檔案，它的內容如下：

```
CHAPTER I. Down the Rabbit-Hole

Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, 'and what is the use of a book,' thought Alice 'without pictures or
conversation?'

So she was considering in her own mind (as well as she could, for the
```

You will also find, on the last page of this exam (which you may tear off), a list of 37 commands. Each command has been assigned a 2 character code (A1, B1, … R2).
在試卷最後一頁，你將可以找到 37 個指令，每個指令已經被分派了 2 個字元碼來表示。

For each of the outputs shown below, select all of the commands that would produce that output. **It must match exactly**. It is possible that none of the commands will generate this precise output – in this case the answer would be R2. But it is also possible that more than one command would generate the given output – in this case the answer would be to list ALL matching commands.
在下方的每一個輸出結果，你必須選出產生與下方完全一樣結果的指令，有可能沒有任何一道指令符合，在這個例子裡答案是R2。也有可能是超過一道指令符合， 此時必須寫出所有符合的答案。

So. your answer will be a series of codes for each number, such as:
所以你的每一題答案將會是一連串的代碼，如下：
1. A1, B1, C2
2. R2
3. E1, A2
…
20. D1

(Now, of course, I make no comment about whether the above choices are correct. I just want to show you the format of your answer.)
(當然，以上並不一定是正確答案，只是展示出答案的格式。)

1.
```
CHAPTER I. Down he Rabbi-Hole

Alice was beginning o ge very ired of siing by her siser on he
bank, and of having nohing o do: once or wice she had peeped ino he
book her siser was reading, bu i had no picures or conversaions in
i, 'and wha is he use of a book,' hough Alice 'wihou picures or
conversaion?'

So she was considering in her own mind (as well as she could, for he
```

2.

```
HAPTER I. Down the Rabbit-Hole

lice was beginning to get very tired of sitting by her sister on the
ank, and of having nothing to do: once or twice she had peeped into the
ook her sister was reading, but it had no pictures or conversations in
t, 'and what is the use of a book,' thought Alice 'without pictures or
onversation?'

o she was considering in her own mind (as well as she could, for the
```

3.

```
CAPTER I. Down the Rabbit-Hole

Aice was beginning to get very tired of sitting by her sister on the
bnk, and of having nothing to do: once or twice she had peeped into the
bok her sister was reading, but it had no pictures or conversations in
i, 'and what is the use of a book,' thought Alice 'without pictures or
cnversation?'

S she was considering in her own mind (as well as she could, for the
```

4.

```
CHAPTER I. Down the Rabbit-Hole
Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, 'and what is the use of a book,' thought Alice 'without pictures or
conversation?'
So she was considering in her own mind (as well as she could, for the
```

5. note: this output is the same as the original input

```
CHAPTER I. Down the Rabbit-Hole

Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, 'and what is the use of a book,' thought Alice 'without pictures or
conversation?'

So she was considering in her own mind (as well as she could, for the
```

6.

```
3:Alice was beginning to get very tired of sitting by her sister on the
6:it, 'and what is the use of a book,' thought Alice 'without pictures or
```

7.

```
1:CHAPTER I. Down the Rabbit-Hole
2:
3:Alice was beginning to get very tired of sitting by her sister on the
4:bank, and of having nothing to do: once or twice she had peeped into the
5:book her sister was reading, but it had no pictures or conversations in
6:it, 'and what is the use of a book,' thought Alice 'without pictures or
7:conversation?'
8:
9:So she was considering in her own mind (as well as she could, for the
```

8.
```
he Rabbit-Hole

o get very tired of sitting by her sister on the
hing to do: once or twice she had peeped into the
er was reading, but it had no pictures or conversations in
, 'and what is the use of a book,' thought Alice 'without pictures or
ion?'

he
```

9.
```
the Rabbit-Hole
the
the
the use of a book,' thought Alice 'without pictures or
the
```

10.
```
1:the
3:her
3:the
4:she
4:the
5:her
6:the
9:she
9:her
9:she
9:the
```

11.
```
book
her
sister
was
reading
but
it
had
no
pictures
or
conversations
in
```

12.
```
4:int
5:ict
6:ict
```

13.
```
1:it
3:it
4:int
5:it
5:ict
6:it
6:it
6:ict
```

14.
```
3
6
```

15.
```
1:abbit
3:as
4:ank
4:and
4:aving
4:ad
5:as
5:ading
5:ad
5:ations
6:and
6:at
6:a
7:ation
9:as
9:as
9:as
```

16.
```
4:an
6:an
9:as
9:as
```

17.
```
3:sitt
5:pictu
6:with
6:pictu
```

18.
```
1:Rabbit
3:sitting
4:into
5:it
5:pictures
6:it
6:without
6:pictures
```

19.
```
2:
8:
```

20.
```
4:and
6:and
6:a
9:as
9:as
```

Part 2. Pattern representations and quoting

In this section, three commands are considered: grep, egrep, and ls. In the various questions below, I will specify a pattern that I want to search for, using one of these three programs. Your job is to decide upon the right pattern, and then locate it from the table shown on the last page. You then write down the corresponding 2-character code (A3, B3,… V7) as the answer to the question.
在這個部分，在下面的不同問題裡有三個指令被考慮使用，使用三個指令其中的一個去搜尋我指定的pattern，你的工作就是決定哪個pattern是對的(從最後一頁的表格裡選擇)，然後寫下其代碼當作問題的答案。

In some cases, none of the answers is correct. Then you have two remaining choices: T7 indicates that the described pattern is not obtainable for the specified command, whereas V7 indicates that the pattern is obtainable, but was not provided within the list. In this case, you must write the correct pattern on the line. Please note: it is incorrect to choose the V7 case if one of the given patterns does work (even if the alternative that you provide also matches the pattern).
在某些例子裡，沒有答案是正確的，那麼你仍然有兩個選擇，T7表示沒有pattern能夠被指定的指令使用。而V7表示爲pattern是存在的但並不在表格裡，此時你必須在答案行上寫出正確的pattern。注意：如果表格中有正確的pattern你卻寫V7，這不會給分，即使你寫的pattern與表格中的一樣。

Let me give you an example. If I had asked you to provide a pattern to list all files in the directory, then you would say:
例子：如果我要求你列出所有目錄下的檔案：
V7 *
You say this because * is the right pattern, but it was not listed in the table. Now, if you had just answered with a V7, you would still get some points, but not all the points.
你寫這答案是因爲，*是正確的pattern且沒有列在表格中。如果你只有答V7只會得到部分分數。

Now, let's make it more complicated! Sometimes the pattern is right, but it needs quotes. Notice that most of the patterns in the table don't have quotes. But grep patterns usually need them. So, if you need a pair of quotes around one of the pattern choices, then you put quotes, in your answer, around that choice. As in 'A3' or "A3".
現在來讓問題複雜點！有時候pattern是對的，但須要引號。注意到大部分在表格中的都沒有引號，但是grep的pattern通常需要。所以你必須在你的答案pattern加上引號。像是'A3' 或 "A3"

Sometimes ls also needs quotes. For an example, consider that choice T5 is ?. If I asked you to list only a file named ? – that is to say, the filename is an actual question mark character – then ls ? will not work. We would need ls '?' or ls "?". So the correct answer, in this case would be 'T5' or "T5".
有時候 ls 也需要引號。例如：選項T5是？但如果我要求你列出只有一個的檔案名爲"？"的呢？也就是說檔名爲一個問號，然後輸入 ls？ ，顯示無法執行。我們必須輸入ls '?' 或 ls "?" 呢？ 所以這個例子的正解是 'T5' 或 "T5"

Now there are five cases:
1. **No quotes allowed for proper behavior** – You will lose half of the points if you put quotes in.
   (Example: If asked to use ls to list all one-character filenames, but you give the non-working answer: 'T5')
如果不需要加引號的你卻加了，你會失去大部分的分數。
範例：我要求使用 ls 列出所有一個字元檔名的檔案，但你卻給了無法執行的答案：'T5'

2. **No quotes needed** – In this case, you will lose a few points if you use them unnecessarily.
   (Example: If asked to use grep to find the number 1, which is R5, but you put unnecessary quotes around it: "R5")
不需要引號 — 在這個例子裡，如果你使用不必要的引號，你將失去一些分數。
範例：如果要求你使用grep去搜索數字1，代碼是R5，但是你多加了引號在代碼的附近："R5"

3. **A quote is needed, but either the single quote or the double quote is acceptable** – You will lose half of the points if you do not use a quote. But you can use either quote that you like.
   (Example: How about the example from above, where you are looking for a file named ?.)
當單引號或是雙引號是必須的，而你卻沒有使用引號的話，你將會失去大部分分數。
範例：尋找檔案名稱爲問號時

4. A single quote is needed – You will lose half of the points if you use no quote or you used a "
當單引號是必須的 — 如果你沒有使用單引號或是使用雙引號，你將會失去大部分分數。

5. A double quote is needed – You will lose half of the points if you use no quote or you used a ’
當雙引號是必須的 — 如果你使用單引號或是沒有使用雙引號，你將會失去大部分分數。

Some notes:

1. You are to assume a real UNIX system, not a Cygwin system with its weird bug about upper-case and lower-case letters.
   假設你所在的環境是UNIX系統，而不是Cygwin。(不會遇到一些奇怪的問題)
2. You are to assume C shell (or, equivalently, tcsh).
   假設你是使用C shell 或是 tcsh
3. You are to choose only one answer. If there is more than one match (and in some cases there are), then you can choose either one. But if you choose two and one of them is wrong then the answer is wrong – so, only choose one.
   答案只有一個。如果有一個以上的符合(有些例子是)，則你必須選其中一個。但如果你選了兩個，其中之一是錯的，那這題妳不會有分數。所以，選一個就好
4. You may NOT use any flags. So, no grep –i, or grep –w, or grep –E, etc.
   你將不會使用到任何的參數。例如：grep －E
5. You may NOT do an ls `____` to change the behavior of ls.
   你不能使用 ls`` 去改變ls的行爲
6. If your pattern for ls has the potential to cause a filename to print twice, then that is NOT a correct answer (because it wouldn’t precisely match the desired output).
   如果你的pattern 導致重複印出檔案名稱兩次，那這個答案是錯的，因爲他不符合我們的需求。
7. In the text below, words are here defined as the maximal-length strings of only letters.
   And fields are defined as the maximal-length strings of anything other than a tab or a space.
   在下面的文字，"words"是定義爲字元組成的最長字串
   而"fields"是定義爲由任何東西除了tab或空白組合的最長字串

21. An **ls** pattern to find all filenames that contain **a period**

22. A **grep** pattern to find all lines containing **a period**

23. An **egrep** pattern to find all lines containing **a period**

24. A **grep** pattern to find all lines containing **a question mark**

25. An **egrep** pattern to find all lines containing **a question mark**

26. An **ls** pattern to find all filenames that **begin with an A or a B**

27. An **ls** pattern to find all filenames that **begin with an A** and **end with a C**

28. An **ls** pattern to find all filenames that **begin with an A** and **end with a C**
or that **begin with an B** and **end with a D**

29. An **ls** pattern to find all filenames that **begin with an A** and **end with a C**
or that **end with a D**

30. A **grep** pattern to find all lines that **begin with an A or a B**

31. A **grep** pattern to find all lines that **begin with an A** and **end with a C**

32. A **grep** pattern to find all lines that **begin with an A** and **end with a C** or that **begin with an B** and **end with a D**

33. A **grep** pattern to find all lines that **begin with an A** and **end with a C** or that **end with a D**

34. An **egrep** pattern to find all lines containing words that **begin with an A or a * symbol**

35. A **grep** pattern to find all lines containing words that **begin with an A** and **end with a C**

36. A **grep** pattern to find all lines containing words that **begin with an A** and **end with a C** or that **begin with an B** and **end with a D**

37. An **egrep** pattern to find all lines containing words that **begin with an A** and **end with a C** or that **begin with anything and end with a ?**

38. An **ls** pattern to find all filenames that contain **at least 2 words** (according to the definition we are using here for a word.)

39. A **grep** pattern to find all lines containing **at least 2 words** (according to the definition we are using here for a word.)

40. An **ls** pattern to find all filenames that contain **a word that repeats twice in the filename** (and separated by 1 non-letter)

41. An **ls** pattern to find all filenames that contain **a word that repeats twice in the filename** (but not necessarily next to each other)

42. A **grep** pattern to find all lines that contain **a word that repeats twice in that line** (but not necessarily next to each other)

43. A **grep** pattern to find all lines that contain **a word that repeats twice in that line but with atleast one word in between the instances of the repeated word**

44. A **grep** pattern to find all lines that contain **two words that repeats twice in that line, in a specific order: word1 … word2 … word2 … word1.** (but not necessarily next to each other)

45. An **egrep** pattern to find all lines that contain **two words that repeats twice in that line, in a specific order: word1 … word2 … word2 … word1.** (but not necessarily next to each other)

46. A **grep** pattern to find all lines that contain **two words that repeats twice in that line, in any order** (but not necessarily next to each other)

47. An **egrep** pattern to find all lines that contain **two words that repeats twice in that line, in any order** (but not necessarily next to each other)

48. A **grep** pattern to find all lines that contain **a word which is exactly 4 letters long and which contains an equal number of A's and B's – but where the A's have to come before the B's.** For example, AABB, ACDB, CEAB and XABZ are all matches, but ZAABB is not. (Note: CDEC also matches, because there are 0 (ie, an equal number of) A's and B's.)

49. An **egrep** pattern to find all lines that contain **a word which is exactly 4 letters long and which contains an equal number of A's and B's – but where the A's have to come before the B's.** For example, AABB, ACDB, CEAB and XABZ are all matches, but ZAABB is not. (Note: CDEC also matches, because there are 0 (ie, an equal number of) A's and B's.)

50. A **grep** pattern to find all lines that contain **a simple string of A's followed by B's. This string only has a A's and B's. It does not need to be a whole word. And it can be of any length greater than 1.** For example, AB, AAB, AAABB, etc.

51. An **egrep** pattern to find all lines that contain **a simple string of A's followed by B's. This string only has a A's and B's. It does not need to be a whole word. And it can be of any length greater than 1. But there must be an equal number of A's and B's.** For example, AB, AABB, AAABBB, etc.

52. An **ls** pattern to find all filenames that contain **at least one digit** within the name

53. An **ls** pattern to find all filenames that contain **exactly one digit** within the name

54. An **grep** pattern to find all lines that contain **at least one digit** within the line

55. An **grep** pattern to find all lines that contain **exactly one digit** within the line

56. An **grep** pattern to find all lines that contain **no more than one digit** within the line

57. An **egrep** pattern to find all lines that contain **no more than one digit** within the line

58. An **grep** pattern to find all lines that contain **two words separated by a number**
For example: cat3dog, A A A0B D, Z2349X, etc.

59. An **ls** pattern to find all filenames that contain **at least one non-letter** within the name

60. A **grep** pattern to find all lines that contain **a + symbol** within the line

61. An **ls** pattern to find all filenames that are **exactly one character long**

62. An **egrep** pattern to find all lines tha**t are exactly one character long**

63. An **ls** pattern to find all files that are **named Alice** – but you must be case insensitive

64. An **egrep** pattern to find all lines that contain **the string Alice** – but you must be case insensitive

65. An **egrep** pattern to find all lines that contains **one of these strings: alice, Alice, or ALICE**.

66. An **ls** pattern to list, if present, a file which is **named as '** (one character, the single quote symbol)

67. An **egrep** pattern to list all lines that contain **a ' symbol** (the single quote symbol)

68. An **ls** pattern to list all filenames that contain **a string of the following four symbols: ?'"*** within the name, and in this order.

69. An **egrep** pattern to list all lines that contain **a string of the following four symbols: ?'"*** within the line, and in this order.

70. An **ls** pattern to find all filenames **ending with whatever is in variable A** (you can assume that variable A holds a string of plain letters)

71. An **egrep** pattern to find all lines **ending with whatever is in variable A** (you can assume that variable A holds a string of plain letters)

Part 3. Short Answers

In this part, you run a UNIX piped command in this format:
在這部分你執行這格式的UNIX管線命令：
% cat filename | _____ > outfile

Your job is to fill in the blank above with a single UNIX command to accomplish each of the following tasks.
你的工作是在空白處填上單一道UNIX指令去完成以下任務.

72. To capitalize all letters in the file

73. To remove all non-letters from the file

74. To display the second word in each line of the file, given that words are separated by spaces

75. To copy the file to another file named "filecopy"

76. To display only the first certain-number-of-lines of the file, where that certain-number-of-lines is to be computed as the sum of two predefined variables, x and y.

77. To insert a space after every symbol in the file.

78. To replace every character in the file with the letter X.

79. To print the number of lines in the file.

Part 4. C-shell programs

80. Write a program which asks the user for 2 numbers and then returns their sum as the $? value.

81. Write a program to list arguments in reverse order, on one line, with a \n at the end.
For example:

```
% ./myprog  A  B C  D
./myprog D C B A
%
```

Note that there might be any number of arguments, including zero:

```
% ./myprog
./myprog
% ./myprog A B C D E F G H I J K L M N O P Q R S T
% ./myprog T S R Q P O N M L K J I H G F E D C B A
```

Note also that you do not know that the name of the program is necessarily myprog. After all, we would expect this to work, right?

```
% cp myprog prog2
% chmod u+x prog2
% ./prog2  A  B C  D
./prog2 D C B A
%
```

Answer Choices

for Part 1(front of this page) and Part 2(back of this page)

A1. `cat file | grep -n "Alice" | sed 's/:.*//'`
B1. `cat file | grep -nC1 "Alice"`
C1. `cat file | grep -win "Alice"`
D1. `cat file | sed -n 's/.*the/the/p'`
F1. `cat file |cut --complement -c 1`
G1. `cat file |cut --complement -d "t" -f 1`
H1. `cat file |cut -f 1`
J1. `cat file |sed 's/.//'`
K1. `cat file |sed 's/.//2'`
L1. `cat file |sed -n 's/./&/p'`
M1. `cat file |tr -d "t"`
N1. `grep "." file`
P1. `grep -n '$' file`
Q1. `grep -n '^' file`
R1. `grep -no he file`
S1. `grep -now '[A-Za-z]*he[A-Za-z]*' file`
T1. `grep -now he file`
V1. `grep -nv '^.+$' file`
W1. `grep -nv '^..*$' file`
X1. `grep -on "\<a[a-zA-Z]" file`
Y1. `grep -on "\<a[a-zA-Z]*" file`
Z1. `grep -on '[a-zA-Z]*i(c|n)*t[a-zA-Z]*' file`
A2. `grep -on '[a-zA-Z]*i[nc]*t[a-zA-Z]*' file`
C2. `grep -on '[a-zA-Z]*ic*t[a-zA-Z]*' file`
D2. `grep -on '[a-zA-Z]ic*t[a-zA-Z]' file`
F2. `grep -on 'a[a-zA-Z]*' file`
G2. `grep -on 'i[nc]*t' file`
H2. `grep -on 'i[nc]+t' file`
J2. `grep -on 'i[nc]?t' file`
K2. `head -5 file | tail -1 | grep -o "[^ ]*[a-zA-Z][^ ]*"`
M2. `head -5 file | tail -1 | grep -o "\<[a-zA-Z]*\>"`
N2. `head -5 file | tail -1 | grep -o "\<[a-zA-Z]+\>"`
P2. `head -5 file | tail -1 | grep -o "\<[a-zA-Z][a-zA-Z]*"`
Q2. `head -5 file | tail -1 | grep -o "\<[a-zA-Z]\>*"`
R2. None of the above

A3- $$A
B3- A
C3- $[A$]
D3- '
F3- ([^A-Za-z][AB])|(^[AB])
G3- (\<A)|(\<B)
H3- (^A.*C$)|(D$)
J3- (alice)|(ALICE)
K3- *$A
L3- *$A*
M3- *+*
N3- *+**
P3- *.**
Q3- *0-9*
R3- *1*
S3- *[$A]
T3- *[.]*
V3- *[.]**
W3- *[0-9]*
X3- *[0-9]?*
Y3- *[0-9]\{0,1]}*
Z3- *[?]['][''][*]*
A4- *[A-Za-z]*.*[A-Za-z]*
B4- *[A-Za-z]*[A-Za-z]*
C4- *[A-Za-z]*[^A-Za-z]*[A-Za-z]*
D4- *[A-Za-z]*[^A-Za-z].*[A-Za-z]*
F4- *[A-Za-z]+.*[A-Za-z]+*
G4- *[A-Za-z].*[A-Za-z]*
H4- *[A-Za-z].*[^A-Za-z][A-Za-z]*
J4- *[A-Za-z]?*[A-Za-z]*
K4- *[A-Za-z]?.*[A-Za-z]?*
L4- *[A-Za-z]?[A-Za-z]*
M4- *[A-Za-z][^A-Za-z]*[A-Za-z]*
N4- *[A-Za-z][^A-Za-z].*[A-Za-z]*
P4- *[Aa][Ll][Ii][Cc][Ee]*
Q4- *[^A-Za-z]*
R4- *[^A-Za-z]?*
S4- *[^A-Za-z][^A-Za-z]*
T4- +
V4- .
W4- .*
X4- .*0-9*.*
Y4- .*0-9.*
Z4- .*1.*
A5- .*[$A]
B5- .*[0-9]*.*
C5- .*[0-9].*
D5- .*[0-9]?.*
F5- .*[0-9][0-9]*.*
G5- .*[0-9][^0-9]*
H5- .*[0-9][^0-9]*.*
J5- .*[0-9]\{0,1]}.*
K5- .*[Aa][Ll][Ii][Cc][Ee].*
L5- .*[^0-9][0-9][^0-9]*.*
M5- .*[a-zA-Z][0-9][0-9]*[a-zA-Z].*
N5- .*[a-zA-Z][0-9][a-zA-Z]*.*
P5- .?
Q5- 0-9
R5- 1
S5- <\A*C|\?\>
T5- ?
V5- ?'"*
W5- ?*0-9?*
X5- ?*1?*
Y5- ?*[0-9]?*
Z5- ?*[0-9]??*
A6- ?*[0-9]\{0,1]}?*
B6- A$
C6- A$$
D6- A*
F6- A*C
G6- A*C D
H6- AA*
J6- AB|AABB|AAABBB
K6- A{A}
L6- [$A]$
M6- [.]
N6- [.]
P6- [.]*
Q6- [.]?
R6- [0-9]
S6- [A-Za-z]*.*[A-Za-z]*
T6- [A-Za-z]*[A-Za-z]
V6- [A-Za-z]*[^A-Za-z]*[A-Za-z]*
W6- [A-Za-z]*[^A-Za-z].*[A-Za-z]*
X6- [A-Za-z]+.*[A-Za-z]+
Y6- [A-Za-z].*[A-Za-z]
Z6- [A-Za-z].*[^A-Za-z][A-Za-z]
A7- [A-Za-z]?*[A-Za-z]
B7- [A-Za-z]?.*[A-Za-z]?
C7- [A-Za-z]?[A-Za-z]
D7- [A-Za-z][^A-Za-z]*[A-Za-z]
F7- [A-Za-z][^A-Za-z].*[A-Za-z]
G7- [AABB]
H7- [AABB]*
J7- [AB]*
K7- [AB]|[AABB]|[AAABBB]
L7- [Aa](lice|LICE)
M7- [^0-9]*[0-9][^0-9]*
N7- [^0-9]*[0-9]\{0,1]}[^0-9]*
P7- [a-zA-Z]*[0-9]+[a-zA-Z]*
Q7- [a-zA-Z]*[0-9]?[a-zA-Z]*
R7- [a-zA-Z]*[0-9][0-9]*[a-zA-Z]*
S7- [A..B]|[A.B.]|[AB..]|[.A.B]|[.AB.]|[..AB]|[AABB]
T7- It is impossible
V7- It is possible, but none of the above work. The correct way is:_____