Java 版の Grep

UNIX の grep コマンドは、正規表現にマッチする行をファイルから抽出します。Windows のコマンドプロンプトで日本語をサポートする grep が必要である場合、Java 版の Grep を試すことができます。ソースコードを添付しました。

java Grep "正規表現" file_name

FINDSTR コマンド

grep に類似のコマンドとして、FINDSTR コマンドが Windows に用意されていますが、コマンドのパラメータが異なるため、FINDSTR は、grep ではありません。

DIR コマンド

Java 版の Grep は、Windows の DIR コマンドを実行して、検索対象のファイルのリストを作成します。ファイル名のリストを fileName.txt と命名します。

ls コマンド

Grep のパラメータとして、-u が付加された場合、Java 版の Grep は、ls コマンドを実行して、fileName.txt を作成します。ls コマンドは、UNIX に用意されています。Windows のパソコンに、たとえば、MinGW をインストールした場合、Windows 用の ls コマンドもインストールされます。コマンドプロンプトで、どのディレクトリでも ls コマンドを利用するには、パスを通す必要があります。

事前に用意

事前に利用者が fileName.txt を用意した場合、しかも Grep のパラメータとして、-b-が付加された場合、Java 版の Grep は、DIR コマンドも ls コマンドも実行しないで、利用者提供の fileName.txt を使用します。

各ファイル名に文字コードを記入

grep と異なり、Java 版の Grep は、fileName.txt の各行から文字コードを捕捉できます。事前に、利用者は、検索対象のファイルのリスト（fileName.txt）の各ファイル名に文字コードを記入できます。

ファイル名の正規表現

DIR コマンドではなく ls コマンドを利用する場合、利用者は、UNIX のシェルのワイルドカードを使用できますが、ファイル名の正規表現も使用できます。

ワイルドカードを正規表現に変換

Java 版の Grep は、ワイルドカードを正規表現に変換して、fileName.txt を作成できま

す。変換結果として、正規表現を表示できます。

カレットおよび英字で制御文字を表現

バイナリファイルに基づいて新しいテキストファイルを作成する機能で、Java 版の Grep は、カレット(^)および英字で制御文字を表現できます。たとえば、01h を^A に置換できます。正規表現にマッチする行を新しいテキストファイルから抽出します。

バイナリファイルを除外

バイナリファイルに基づいて新しいテキストファイルを作成しないで、Java 版の Grep は、バイナリファイルであるかどうか判定して、バイナリファイルを除外して、次のファイルに着手できます。

ACE にリダイレクト

コマンドプロンプトのウィンドウとは独立した SideAce ウィンドウが表示されます。SideAce は、ACE を監視しています。リダイレクトで ACE に書き込まれたとき、SideAce ウィンドウに内容を表示します。SideAce は、ACE を UTF-8 で保存できます。

インストール

Java 版の Grep は、Java のアプリケーションです。まず Java をインストールしてください。Java のバージョン番号は、1.5 以上であることが必要です。次に、Grep をインストールしてください。

手順

1. トリシーカー(http://tori.tobiiro.jp/)というウェブサイトから GrepJava.zip をダウンロードしてください。
2. GrepJava.zip を展開してください。
3. たとえば、C:¥Grep¥info¥lotosummary¥software¥GrepError.class となるように、Grep というフォルダを移動してください。フォルダのことをディレクトリとも言います。
4. Windows ロゴキー(田キー)を押しながら、Pause を押してください。
5. Windows Vista, Windows 7 の場合、メニューからシステムの詳細設定を選択してください。
6. 環境変数ボタンを押してください。キーワード:**システム、詳細、環境変数**
7. システム環境変数として、CLASSPATH を選択して、[編集]ボタンを押してください。**CLASSPATH が無い場合、[新規]ボタンを押してください。**
8. Windows 10 の場合、[テキストの編集]ボタンを押してください。

9. 変数名として CLASSPATH を入力してください。
10. 右向き矢印キー(→)または End キーを押してください。カーソルが末尾に移動します。セミコロンは、ディレクトリどうしを区切る記号です。
11. 変数値として、ドット、セミコロン、C:¥Grep が含まれるように入力してください。ドットは、カレントディレクトリを意味します。

```
.;C:¥Grep
```

12. 各ウィンドウで[OK]ボタンを押してください。

アンインストール

アンインストール(プログラムの削除)を行うには、フォルダ(C:¥Grep)ごと削除してください。CLASSPATH から C:¥Grep を削除してください。

著作権

Sogaya(そがや)は、Java 版の Grep の著作権を保有しています。

免責条項

Java 版の Grep のご利用によって発生するいかなる損害も、Sogaya は、責任を負わないものとします。

JDK 1.8

32 ビット版 Windows 7 のパソコンに JDK 1.8 をインストールした環境で Java 版の Grep を構築しました。ソースコードを添付しました。

構築

32 ビット版 Windows XP のパソコンに JDK 1.5 をインストールした環境で同じソースコードをコンパイルして、Java 版の Grep を構築できます。64 ビット版 Windows 10 のパソコンに JDK 1.8 をインストールした環境で同じソースコードをコンパイルして、Java 版の Grep を構築できます。Java 版の Grep は、正規表現および ArrayList<String>を使用するため、1.5 以上のバージョン番号を有している Java が必要です。【課題】添付の SideAce は、大きいウィンドウが Windows XP に適していないと思われます。

参考

Java 版の Grep に基づいて、C# で AceGrep を構築しました。トリシーカー(http://tori.tobiiro.jp/)というウェブサイトから AceGrep.zip をダウンロードできます。

ユーザーズガイド

2018 年 12 月 1 日。Revision 1.032

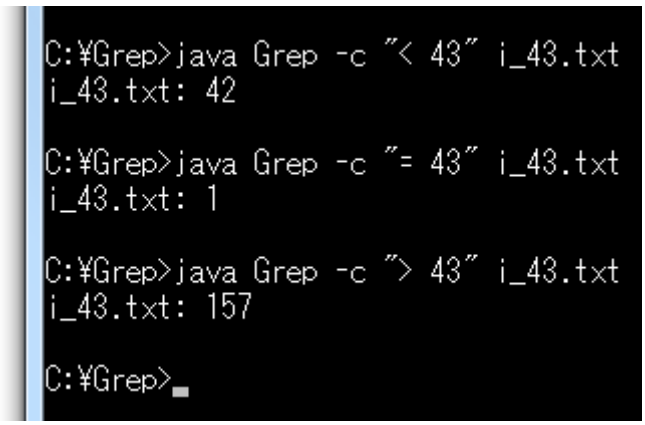
Grep のパラメータ

テストファイルを添付しました。英語のテストファイルが、English というディレクトリにあります。日本語のテストファイルが、Japanese というディレクトリにあります。AWK のスクリプトが、test_files というディレクトリにあります。以下の説明で、こうしたテストファイルを利用します。

ファイル名および行数

UNIX の grep と同様に、Java 版の Grep は、正規表現にマッチした行をファイルから抽出しますが、行を抽出しないで、行をカウントして、ファイル名および行数を出力するには、Grep のパラメータとして、-c を付加してください。下線(_)を入力するには、Shift を押しながら、ろを押してください。

```
java Grep -c "< 43" i_43.txt
```



マッチしない行

マッチしない行をカウントして、ファイル名および行数を出力するには、Grep のパラメータとして、-c および-v を付加してください。v は、小文字です。

```
java Grep -c -v "< 43" i_43.txt
```

排他的論理和

行が正規表現にマッチしないと判定するために、Java 版の Grep は、排他的論理和を利用します。

パラメータ	-v	0	マッチしなかった	排他的論理和	説明
		1	マッチした		
-c	0	0		0	
-c	0	1		1	カウントする
-c -v	1	0		1	カウントする
-c -v	1	1		0	

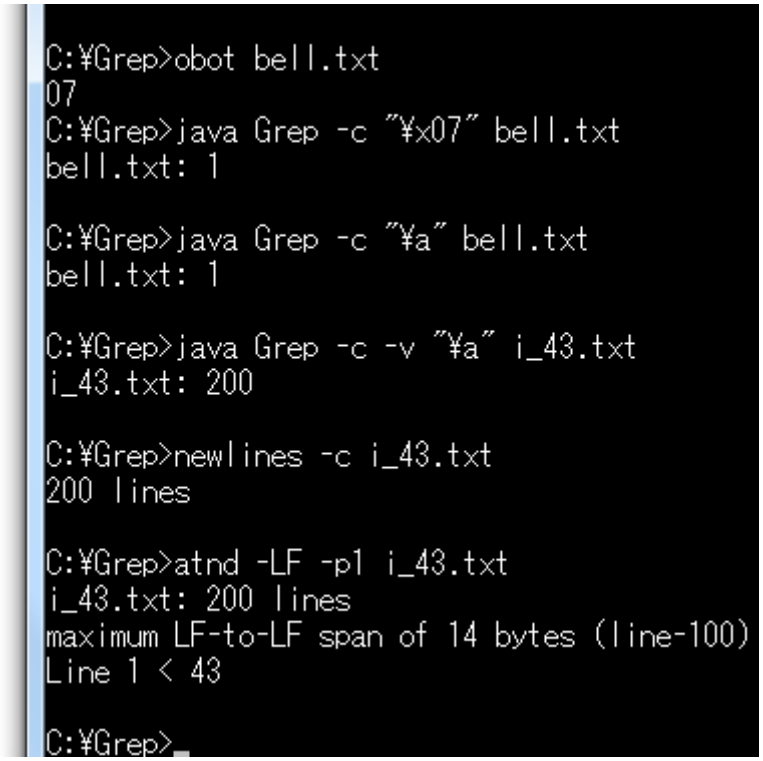
ファイルの行数

どの行にもマッチしない正規表現を入力することで、ファイルの行数を出力できます。

¥a

どの行にもマッチしないと思われる文字としては、ベル文字(07h)があります。正規表現の¥x07 は、十六進数の 07hを意味します。ベル文字は、¥aと表記されますが、1 バイトです。たとえば、bell.awkは、3 バイトのベル文字列を出力するため、ピーという音が 3 回聞こえます。AWKの場合、¥aは、警告(アラート)を意味します。1 バイトの整数として、¥x07 も¥aも 7 に等しい。正規表現として¥aを指定した例

java Grep -c -v "¥a" i_43.txt



obot コマンド

Java 版の Grep に obot コマンドを添付しました。obot コマンドは、ファイルの冒頭を十

六進数の列で表示します。bell.txt は、ベル文字（07h）のみからなるファイルです。すなわち、bell.txt のサイズは、1 バイトです。

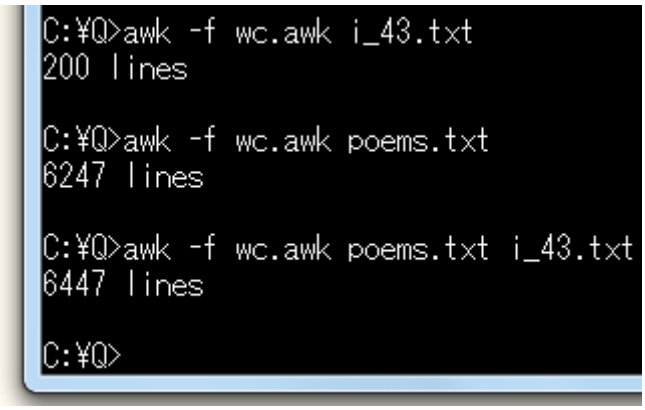
wc コマンド

MinGW をインストールしたら、wc コマンドもインストールされます。GNU Win32 の coreutils-5.3.0 をインストールしたら、wc コマンドもインストールされます。使用法を表示するには、--help を付加してください。UNIX の wc コマンドで、行数を出力するには、コマンドプロンプトに下記のコマンドおよびパラメータ（マイナスエル）を入力して、Enter を押してください。

```
wc -l i_43.txt
```

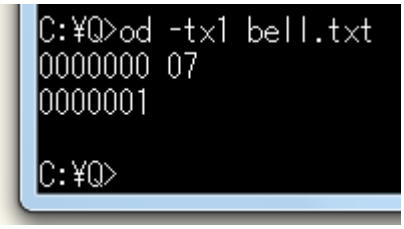
AWK

Java 版の Grep に wc.awk を添付しました。多数のファイルの行数の合計を表示できます。



od コマンド

GNU Win32 にも MinGW にも、od コマンドがあります。bell.txt は、07h のみからなる 1 バイトのテキストファイルです。od でこのことを示した例



newlines コマンド

Java 版の Grep に newlines コマンドを添付しました。newlines コマンドは、2 連の改行を単独の改行（0Ah）に置換します。元のファイルを変更しないで、新しいファイルを作成します。利用者が-c を指定した場合、newlines コマンドは、行数を出力します。ただし、**ファイルの終わりの行に改行文字が無い場合、カウントしません。**

atnd コマンド

Java 版の Grep に atnd コマンドを添付しました。atnd コマンドは、-p で指定された行の周辺を出力します。利用者が-LF を指定した場合、atnd コマンドは、行数、最も長い行の長さ(行番号)、指定された行を出力します。

fileName.txt は削除される

-b-が指定されない限り、DIR コマンドを実行する前に、Java 版の Grep は、既存の fileName.txt を削除します。したがって、DIR コマンドの実行結果が生じるまで、fileName.txt のサイズは、0 です。

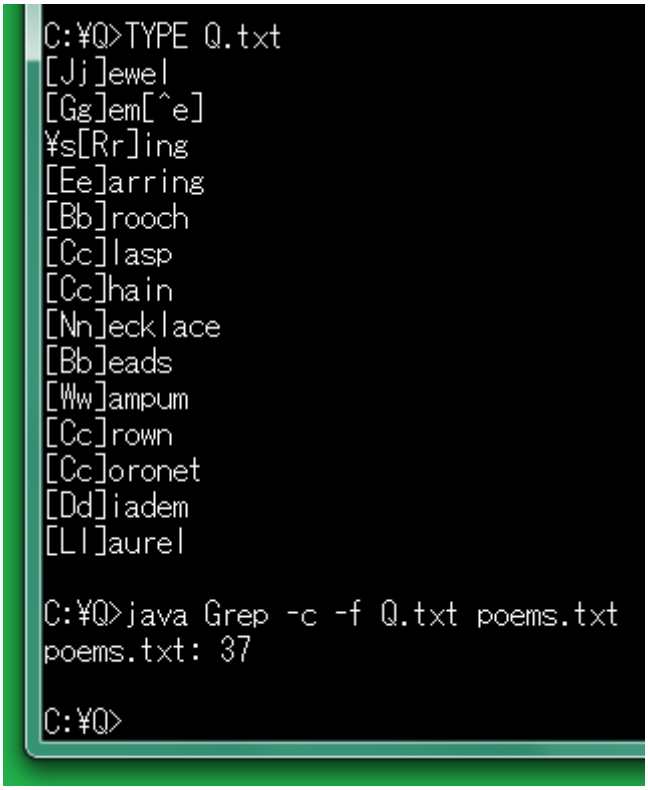
AND 検索

--label=に関して後述しますが、2 個以上の正規表現で、いわゆるAND検索を行うには、パイプを構成してください。さもなければ、AND検索になるように、2 個以上の正規表現から 1 個の正規表現を案出してください。ANDとは、論理積のことです。ORとは、論理和のことです。

多数の正規表現

2 個以上の正規表現で、いわゆる OR 検索を行うには、まず正規表現のリスト(patterns.txt)の各行に正規表現を記入してください。次に、コマンドプロンプトにコマンドを入力して、パラメータとして、-f を入力して、patterns.txt を入力して、パラメータの最後に検索対象ファイルを入力して、Enter を押してください。

```
java Grep -c -f patterns.txt poems.txt
```



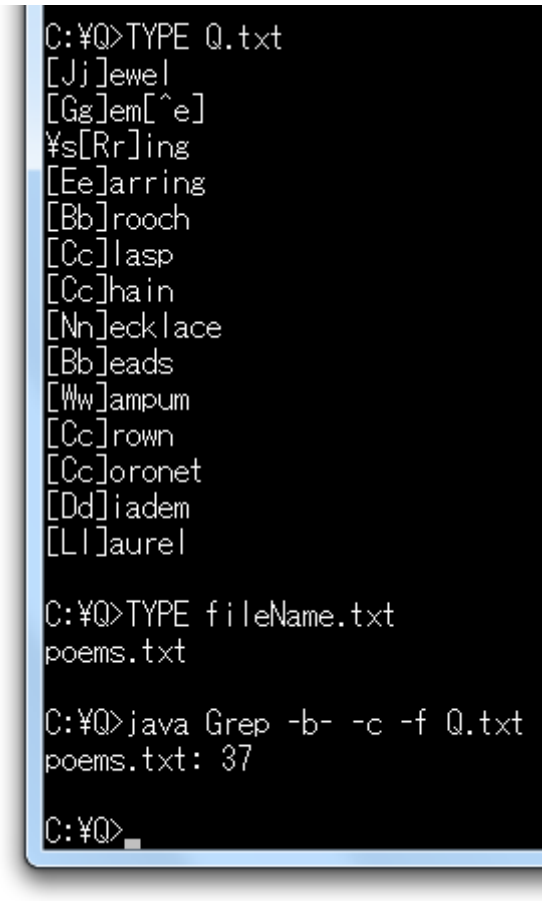
最後に検索対象ファイルを入力

上図の例では、正規表現のリストを Q.txt と命名しています。Java 版の Grep は、コマンドのパラメータの最後にある文字列（poems.txt）を検索対象ファイルとして DIR コマンドにわたします。

最後に正規表現のリストを入力

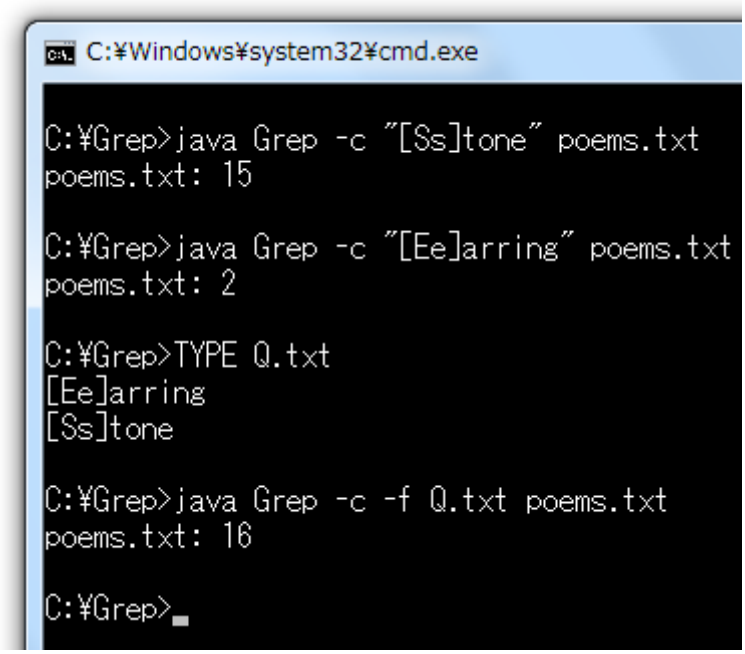
必要な fileName.txt がすでに作成されている場合があります。この場合、検索対象ファイルを入力しないで、正規表現のリストを最後に入力するには、-b-を指定してください。この場合、DIR コマンドは実行されません。

```
java Grep -b- -c -f Q.txt
```



1 行で 2 個以上マッチするか

Java 版の Grep は、少なくとも 1 個一致する行をカウントします。Java 版の Grep は、ある行で 2 個以上マッチするか判定しないで、マッチしたら次の行に着手します。ある行で 2 個マッチすると思われる例を以下に示します。以下の例では、同一の行を 2 重にカウントしたため、個別の正規表現で検索した行数の和は、Java 版の Grep が OR 検索を行った行数より大きい（15 + 2 > 16）。



バージョン番号

Java 版の Grep のバージョン番号を表示するには、コマンドプロンプトに以下のコマンドおよびパラメータを入力して、Enter を押してください。V は、大文字です。

```
java Grep -V
```

正規表現追加モード

正規表現リストではなく多数の正規表現をパラメータとして入力するには、まず-e を入力してください。次に、多数の正規表現を入力してください。ただし、各正規表現を引用符で囲んでください。多数の正規表現が追加されます。

```
java Grep -c -e "[Ss]tone" "[Ee]arring" -e- poems.txt
```

正規表現追加モードを終了

上図で示したとおり、正規表現追加モードを終了するには、-e-を入力してください。Java 版の Grep は、自動的に patterns.txt を作成します。すなわち、OR 検索になります。Java 版の Grep は、コマンドのパラメータの最後にある文字列（poems.txt）を検索対象ファイルとして DIR コマンドにわたします。**検索対象ファイルを入力する前に、-e-を入力してください。**

-e で開始して副作用で終了

多数のパラメータは、それぞれ機能を有していますが、副作用で正規表現追加モードを終了できます。利用者が-e を入力したら、正規表現追加モードになります。-e-ではなく-c でも正規表現追加モードを終了できます。

```
java Grep -e "[Ss]tone" "[Ee]arring" -c poems.txt
```

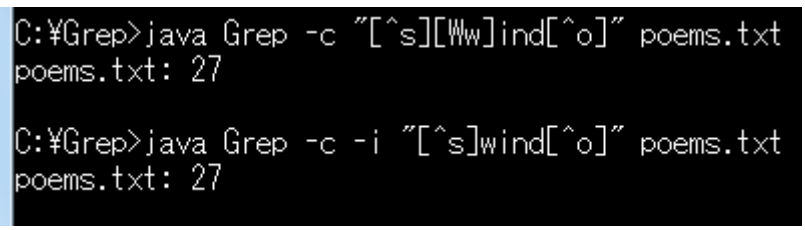
副作用があるパラメータの例

-e-は、正規表現追加モードを終了するために特別に用意されたパラメータです。

```
--ace    --re -8  -g -j  -Ku -n  -S -y
--ascii  -    -b- -H -k  -Kw -p- -u
--ce     -@   -c  -I -Ke -l  -q  -v
--elapse -[   -f  -i -Ks -L  -r  -x
```

大文字と小文字を区別しない

英文で、大文字と小文字を区別しないで検索するには、[Ss], [Ee]などの正規表現を案出する以外にも、パラメータとして-i を付加する方法もあります。i は、小文字です。



一時ファイルを命名する

Java 版の Grep は、Windows の DIR コマンドを実行して、検索対象のファイルのリストを作成して、fileName.txt と命名します。fileName.txt ではない新しいファイルとして、一

時ファイルを命名するには、-j を指定してください。FNL 番号-分-秒.txt の形式で命名されます。serial.txt に記載された数値に 1 を加算した数を番号とします。

通し番号

Java 版の Grep は、FNL 番号-分-秒.txt の一時ファイルを作成するたびに、serial.txt の数値に 1 を加算して、計算結果を serial.txt に書き出します。すなわち、一時ファイルは、通し番号が付与されたファイルです。

ファイル名に時刻がある

利用者が時刻を手掛かりにしてファイルを探しやすいように、Java 版の Grep は、通し番号だけではなく、時刻（分、秒）もファイル名に導入します。

funbyo コマンド

Java版のGrepにfunbyoコマンドを添付しました。WindowsのTIMEコマンドは、/Tのパラメータで現在の時刻の時、分を表示します。funbyo コマンドは、現在の時刻の分、秒を表示します。funbyo コマンドは、-gのパラメータで、年月日時分秒を表示します。funbyo コマンドは、改行しないで終了します。

```
C:¥Q>DEL FNL*.txt

C:¥Q>TYPE serial.txt
8

C:¥Q>TIME /T
20:42

C:¥Q>funbyo
42:32
C:¥Q>java Grep -c -j "[^dfp][Rr]ank" poems.txt
poems.txt: 6

C:¥Q>DIR FNL*.txt   | java Grep 'FNL'
2018/11/25  20:42                11 FNL9-42-32.txt

C:¥Q>
```

rvoy コマンド

Java 版の Grep に rvoy コマンドを添付しました。FNL 番号-分-秒.txt の形式で命名された一時ファイルを削除するには、まず rvoy コマンドを実行してください。ovee コマンドでファイルを削除するために、remove_e.txt が作成されます。

ovee コマンド

Java 版の Grep に ovee コマンドを添付しました。ovee コマンドは、remove_e.txt にしたがって、ファイルを削除します。重要なファイルを削除されないように、ovee コマンドを

実行する前に、利用者は、remove_e.txt から重要なファイルの名前を削除できます。ovee コマンドを実行したのち、どのファイルが削除されたか、remove_e.txt から認識できます。

シングルクォーツ

正規表現を引用符(")で囲んでください。FINDSTR コマンドと異なり、Java 版の Grep は、**シングルクォーツ**(')で正規表現を囲んでも同じ結果を生じます。

```
C:¥Grep>TYPE sayno.txt
What does 'pulchritude' mean?
The little girl pulled gently at my sleeve.
Everything depends on what you mean by the word 'free'
and you are free to come and go as you please.
When you shake your head it usually means 'no'
but you will have to say no this time.

C:¥Grep>java Grep -c "no" sayno.txt
sayno.txt: 2

C:¥Grep>java Grep -c 'no' sayno.txt
sayno.txt: 2

C:¥Grep>java Grep -c "¥'no¥'" sayno.txt
sayno.txt: 1

C:¥Grep>_
```

エスケープ

単語を囲んでいるシングルクォーツを含めるには、¥でシングルクォーツをエスケープしてください。すなわち、シングルクォーツの直前に¥を入力してください。Java 版の Grep は、自動的に¥'をシングルクォーツに置換します。

パイプ

パイプは、コマンドの出力をもうひとつのコマンドに入力して適切に処理します。縦線(|)を使用します。DIR コマンドの出力を FINDSTR コマンドに入力するパイプの例

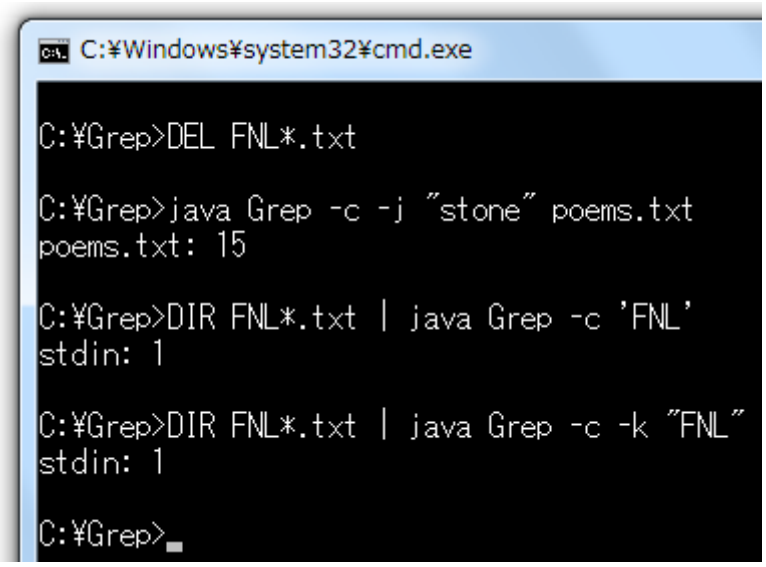
```
DIR FNL*.txt | FINDSTR "FNL"
```

検索対象のファイルを入力しない

検索対象のファイルが入力されなかった場合、FINDSTRコマンドは、標準入力が情報源であるとみなします。検索対象のファイルが入力されなかった場合、Java 版の Grep が標準入力を情報源とみなすように、引用符(")ではなく、シングルクォーツで正規表現を囲んでください。さもなければ、パラメータとして-k を付加してください。k は、小文

字です。

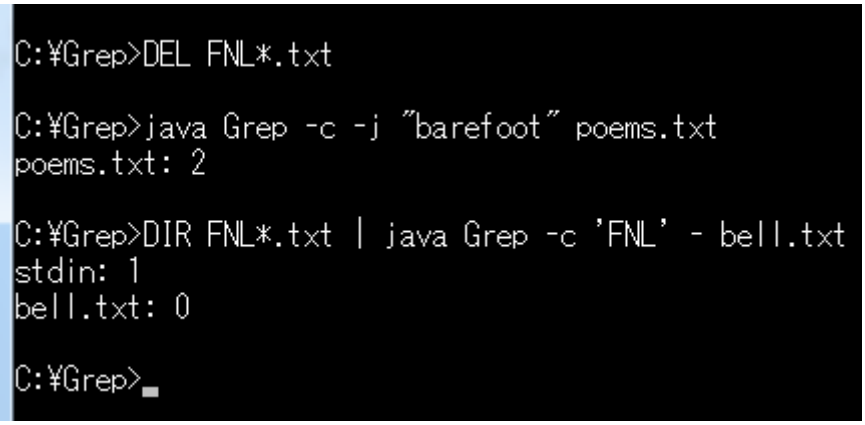
DIR FNL*.txt | java Grep -c 'FNL'



標準入力を意味するハイフン

上図に示されたとおり、stdin は、標準入力を意味します。Java 版の Grep は、stdin がファイルであるかのように処理します。もうひとつのファイルを同時に検索するには、-k ではなく、ハイフン(-)のパラメータを付加してください。

DIR FNL*.txt | java Grep -c 'FNL' - bell.txt



情報源を命名

stdin ではなく情報源を命名したい場合、--label=名前の形式で指定してください。たとえば、stdin ではなく DIR を明示するには、パラメータとして、--label=DIR を付加してください。

```
C:¥Grep>DEL FNL*.txt

C:¥Grep>java Grep -c -j "[Ss]ong" poems.txt
poems.txt: 4

C:¥Grep>DIR FNL*.txt | java Grep -c --label=DIR 'FNL' - bell.txt
DIR: 1
bell.txt: 0

C:¥Grep>_
```

パイプで AND 検索

パイプにより AND 検索が可能です。まず、say を含む行をファイルから抽出して、次に、抽出結果から no を含む行を抽出する例を示します。

```
C:¥Grep>FINDSTR "say" sayno.txt | FINDSTR "no"
but you will have to say no this time.

C:¥Grep>FINDSTR "say" sayno.txt | java Grep -c --label=FINDSTR 'no'
FINDSTR: 1

C:¥Grep>_
```

テキストファイル

テキストエディタで編集しやすいファイルをテキストファイルと言います。テキストファイルの拡張子として、.txt および.csv が多用されます。.csv は、コンマで分離された値からなるテキストファイルの拡張子であり、Excel などの表計算ソフトウェアで.csv のファイルを読み込むことができます。

NOTEPAD コマンド

Windows のメモ帳は、テキストエディタであり、コマンドプロンプトに、NOTEPAD を入力して、Enter を押すことで起動できます。テキストファイルを開くには、NOTEPAD コマンドのパラメータとしてファイル名を入力して、Enter を押してください。

NOTEPAD remove_e.txt

テキストエディタ

無料のテキストエディタとしては、サクラエディタ、TeraPad などがあります。EmEditor，MIFES など、有料のテキストエディタには、期間限定版、試用版、体験版があります。

バイナリファイル

テキストファイルではないファイルをバイナリファイルと言います。編集どころか閲覧す

ら困難であるため、ある文字列を検索するために、バイナリファイルに基づいてテキストファイルを作成したい場合があります。

符号を出力

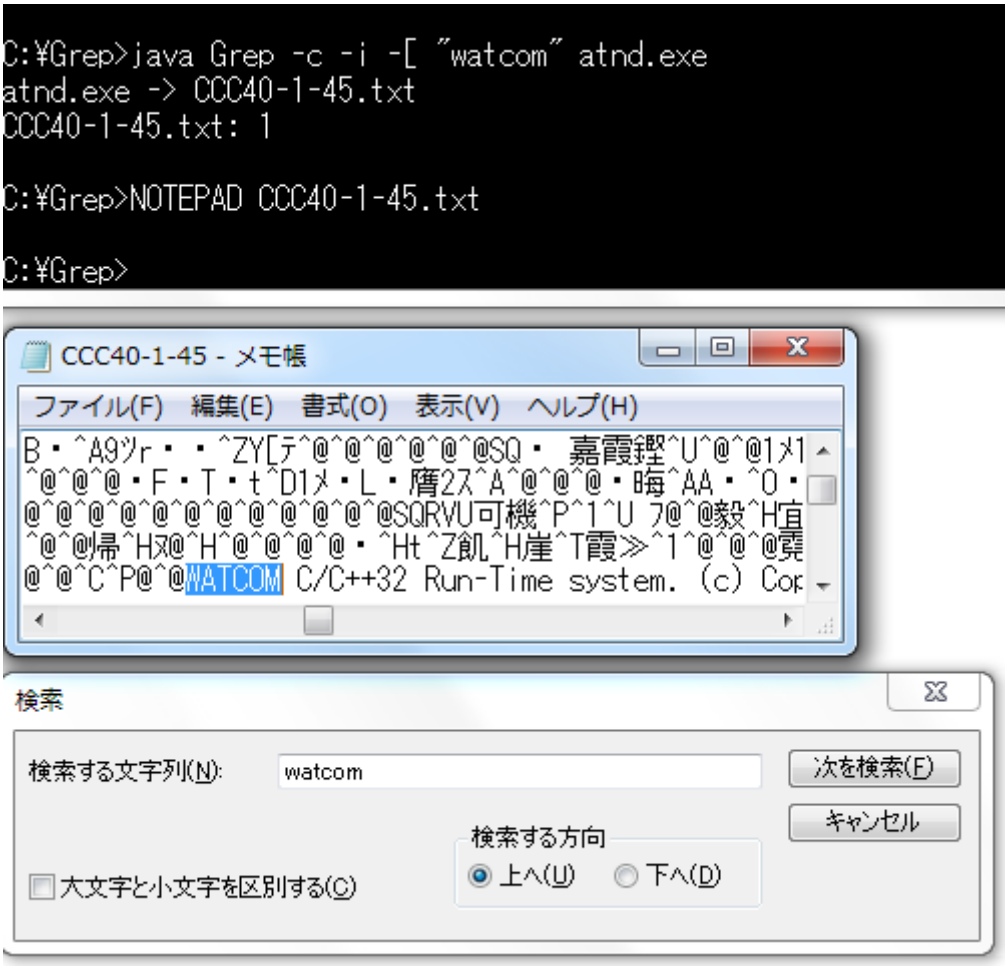
Java 版の Grep は、いわゆる制御文字（数値）ではなく、^L, ^Z, ^[などの符号を出力できます。ただし、制御文字として、復帰、改行、タブは、それぞれ 13, 10, 9 の数値が出力されます。

数値	符号		数値	符号		数値	符号		数値	符号
-3	^3		11	^K		14	^N		127	^7
-2	^2		12	^L		15	^O			
-1	^1					16	^P			
0	^@					17	^Q			
1	^A					18	^R			
2	^B					19	^S			
3	^C					20	^T			
4	^D					21	^U			
5	^E					22	^V			
6	^F					23	^W			
7	^G					24	^X			
8	^H					25	^Y			
						26	^Z			
						27	^[			
						28	^¥			
						29	^]			
						30	^^			
						31	^_			

コンパイラ名

添付の atnd.exe が Watcom で構築されていることを確認するには、パラメータとして、-[を付加してください。**一時ファイル**が CCC 番号-分-秒.txt の形式で命名されます。serial.txt に記載された数値に 1 を加算した数を番号とします。Java 版の Grep は、CCC 番号-分-秒.txt の一時ファイルを作成するたびに、serial.txt の数値に 1 を加算して、計算結果を serial.txt に書き出します。すなわち、一時ファイルは、通し番号が付与されたファイルです。利用者が時刻を手掛かりにしてファイルを探しやすいように、Java 版の Grep は、通し番号だけではなく、時刻（分、秒）もファイル名に導入します。

```
java Grep -c -i -[ "watcom" atnd.exe
```



バイナリファイルなら中止

Java 版の Grep のパラメータとして最後に入力された文字列が DIR コマンドにわたされます。DIR コマンドの結果として、検索対象ファイルのリスト(fileName.txt)が作成されます。fileName.txt に含まれているファイルがバイナリファイルであるかどうか判定して、バイナリファイルであれば中止して、次のファイルに着手するには、パラメータとして、-I(マイナスアイ)を付加してください。I は、大文字です。

4%

冒頭の 2048 バイトまで検査されます。ただし、この判定が正しいことは、保証されません。いわゆる制御文字の例として、1Bh は、エスケープシーケンスで符号化文字集合を切り替えます(状態を変更します。シフトします)。略号は、ESC です。制御文字のうち、復帰、改行、改ページ、タブ、ベルなどの下記の数値は、制御文字ではないとみなします。それにもかかわらず制御文字の個数が総数の 4%より多い場合、Java 版の Grep は、テキストエディタで編集しにくいと判定します。添付の obot コマンドは、ファイルの冒頭を十六進数の列で表示します。

07h 08h 09h 0Ah 0Bh 0Ch 0Dh

十六進数	数値	英語	略号	説明
07h	7	**alert**	BEL	ベル、警告
08h	8	**back space**	BS	バックスペース
09h	9	**horizontal tab**	HT	タブ、水平タブ
0Ah	10	**line feed**	LF	改行
0Bh	11	**vertical tab**	VT	垂直タブ
0Ch	12	**form feed**	FF	改ページ
0Dh	13	**carriage return**	CR	復帰

多数のファイル名を引用符で囲む

Java 版の Grep は、最後に入力された文字列を DIR コマンドにわたします。Java 版の Grep が多数のファイル名を一括して DIR コマンドにわたすように、多数のファイル名を引用符(″)で囲んでください。**DIR コマンドにわたす文字列をシングルクォーツで囲まないでください。**

ワイルドカード

コマンドプロンプトは、ワイルドカードを展開しますが、Java 版の Grep は、コマンドプロンプトに出現したワイルドカードを展開しません。Java 版の Grep がそのまま DIR コマンドにわたすためには、利用者は、**ワイルドカード表現のファイル名**を引用符で囲まなければなりません。

ls コマンド

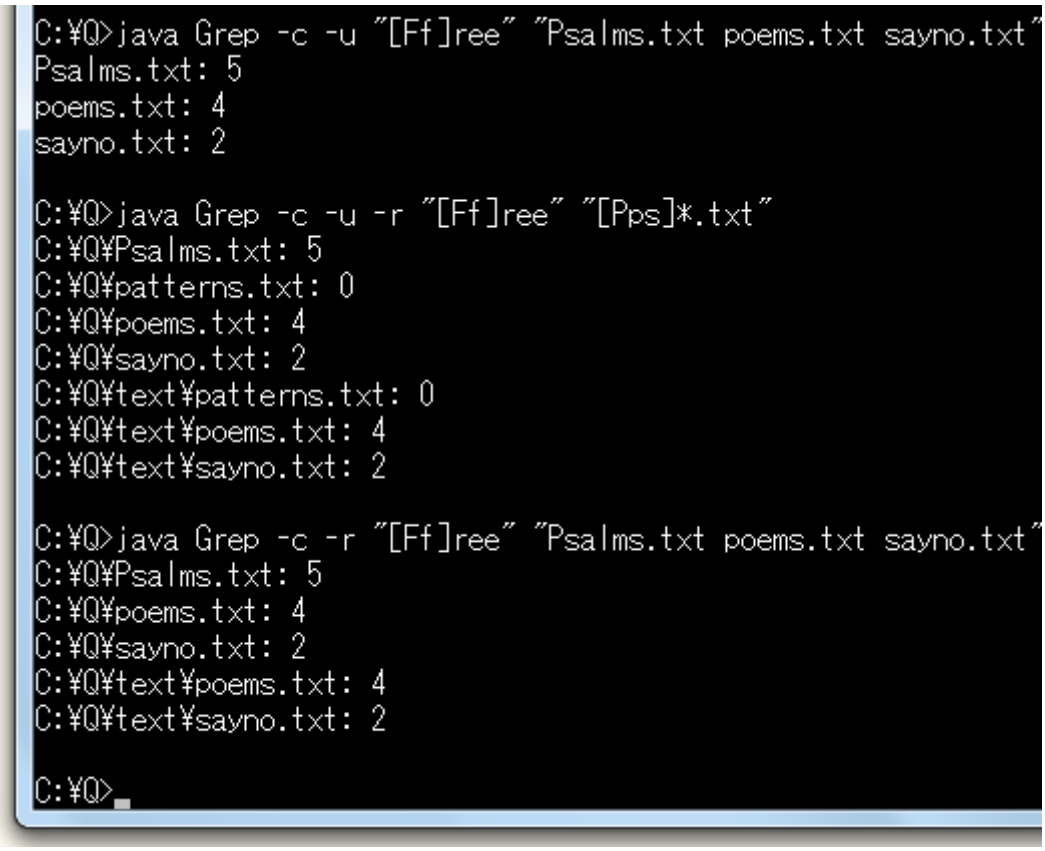
MinGW は、Ada, FORTRAN, C++, Objective-C の開発環境です。Windows のパソコンに、たとえば、MinGW をインストールした場合、Windows 用の ls コマンドもインストールされます。検索対象ファイルのリスト(fileName.txt)を作成するために、DIR コマンドではなく、ls コマンドを利用するには、-u を指定してください。

ファイル名を引用符で囲む

Java 版の Grep がそのまま ls コマンドにわたすように、ワイルドカード表現のファイル名を引用符で囲んでください。Java 版の Grep が多数のファイルを一括して ls コマンドにわたすように、多数のファイル名を引用符(″)で囲んでください。**ls コマンドにわたす文字列をシングルクォーツで囲まないでください。**

サブディレクトリにある多数のファイル

カレントディレクトリにある多数のファイルを引用符で囲んで、-u を指定して ls コマンドにわたすことができますが、-u -r を指定してサブディレクトリも検索する場合、多数のファイルを引用符で囲まないで、シェルのワイルドカードまたは正規表現を使用してください。



サブディレクトリも検索

Java 版の Grep は、fileName.txt に記載の各ファイルを開いて検索します。fileName.txt に下位のディレクトリ（サブディレクトリ）にあるファイルを含めるには、パラメータとして、-r を付加してください。recursive は、数学、コンピュータサイエンスの用語であり、grep の場合、-r は、**再帰的なディレクトリの検索**を意味します。サブディレクトリにもサブディレクトリがあるかもしれません。すなわち、ディレクトリを検索して、サブディレクトリを検索して、次のサブディレクトリを検索して、次々にサブディレクトリを検索して、結果として、カレントディレクトリおよび下位のディレクトリを網羅します。

ワイルドカードを正規表現に変換

DIR コマンドは、コマンドプロンプトの内部コマンドですが、ls コマンドは、シェル（bash, tcsh など）の外部コマンドです。-u および-r が指定された場合、Java 版の Grep は、**シェルのワイルドカード**を正規表現に変換します。ただし、この変換が正しいことは、保証されません。

変換結果としての正規表現

-u および-r が指定された場合、Java 版の Grep は、シェルのワイルドカードを正規表現に変換しますが、シェルのワイルドカードが正規表現に正しく変換されたか確認するには、パラメータとして、--re を付加してください。

java Grep -c -u -r --re "fame" "[!f]*.txt"

```
C:¥Q>java Grep -c -u -r --re "fame" "[!f]*.txt"
Regular expression: ^[^f].*¥.txt$
C:¥Q¥bell.txt: 0
C:¥Q¥rannu.txt: 0
C:¥Q¥text¥Q.txt: 0
C:¥Q¥text¥i_43.txt: 0
C:¥Q¥text¥patterns.txt: 0
C:¥Q¥text¥poems.txt: 6
C:¥Q¥text¥quotes.txt: 0
C:¥Q¥text¥sayno.txt: 0

C:¥Q>
```

正規表現のファイル名

-u および-r が指定された場合、Java 版の Grep は、ワイルドカード表現のファイル名でも正規表現のファイル名でも fileName.txt を作成できます。Java 版の Grep が正規表現のファイル名で fileName.txt を作成するように、斜線(/)で正規表現を囲んでください。さらに、引用符で囲んでください。$は、ファイルの各行の行末を意味します。

java Grep -c -u -r "fame" "/^[^f].*¥.txt$/"

"/^[^f].*¥.txt$/"

ファイル名の大文字小文字を区別しない

検索文字列の場合、大文字を小文字と区別しないパラメータは、-i ですが、検索対象ファイル名の場合、大文字を小文字と区別しないように表現するには、最後に$/"ではなく、**$/i"**を入力してください。

java Grep -c -u -r "fame" "/^[^f].*¥.txt$/i"

カレントディレクトリのみ関心がある

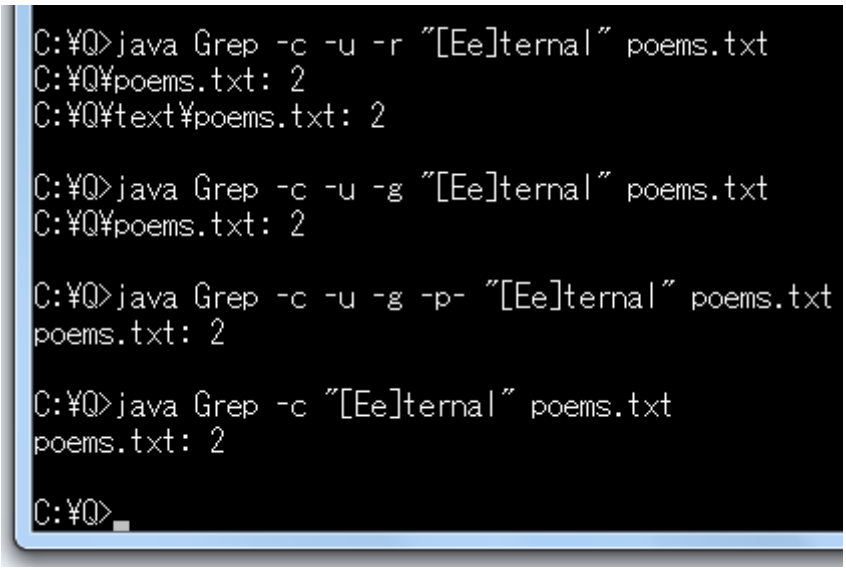
-u および-r が指定された場合、Java 版の Grep は、正規表現のファイル名でも fileName.txt を作成できますが、fileName.txt からサブディレクトリを消去するには、-g を指定してください。カレントディレクトリのみで正規表現のファイル名を使用して検索できます。-r および-g の両方を指定した場合、-g と同じ結果になります。

java Grep -c -u -g "fame" "/^[^f].*¥.txt$/"

パス名を表示しない

-u -g が指定された場合、Java 版の Grep は、カレントディレクトリにあるファイルのパス名を表示しますが、パス名を表示しないように命令するには、パラメータとして、-p-を付加してください。

java Grep -c -u -g -p- "fame" "/^[^f].*¥.txt$/"



-u および-g でワイルドカード

-u および-g が指定された場合、Java 版の Grep は、シェルのワイルドカードを正規表現に変換します。ただし、この変換が正しいことは、保証されません。シェルのワイルドカードが正規表現に正しく変換されたか確認するには、パラメータとして、--re を付加してください。-u -g の場合、多数のファイルを引用符で囲まないで、シェルのワイルドカードまたは正規表現を使用してください。

-u および-r が指定されたら万能か

-u のみである場合、シェルのワイルドカードによりカレントディレクトリを検索できます。-u および-g が指定された場合、シェルのワイルドカードまたは正規表現によりカレントディレクトリを検索できます。下表のとおり、-u および-r が指定された場合、シェルのワイルドカードまたは正規表現によりカレントディレクトリおよびサブディレクトリを検索できます。ただし、-u -r の場合、多数のファイルを引用符で囲んで ls コマンドにわたすことができません。-u -r の場合、多数のファイルを引用符で囲まないで、シェルのワイルドカードまたは正規表現を使用してください。

	シェルのワイルドカード	正規表現
カレントディレクトリ	−u	−u −g
サブディレクトリ	−u −r	

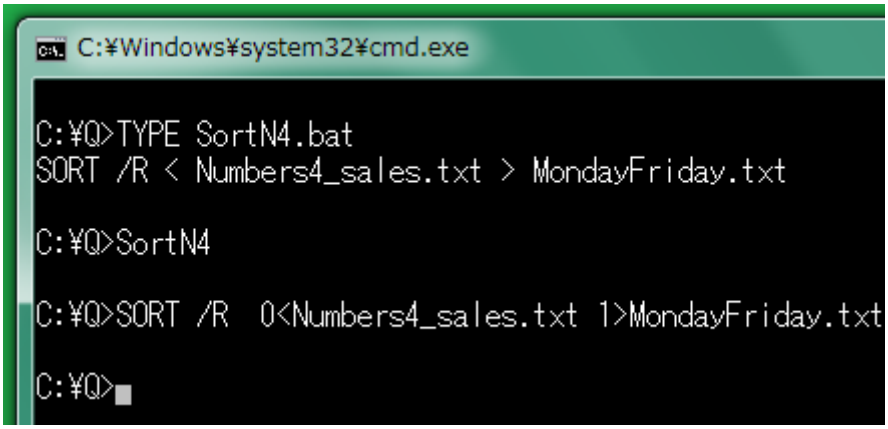
バッチファイル

−u および−r が指定された場合、あるいは、−u および−g が指定された場合、Java 版の Grep は、まず、LS_R_1.bat というバッチファイルを実行します。次に、Java 版の Grep は、ls コマンドの結果として生じた fileName.txt を読み込んで、新しい fileName.txt を書き出します。ls コマンドの結果として生じた fileName.txt は、上書きされます。上書きされる前の fileName.txt を再現するには、コマンドプロンプトに C:¥Grep¥LS_R_1 を入力して、Enter を押してください。

C:¥Grep¥LS_R_1

1>

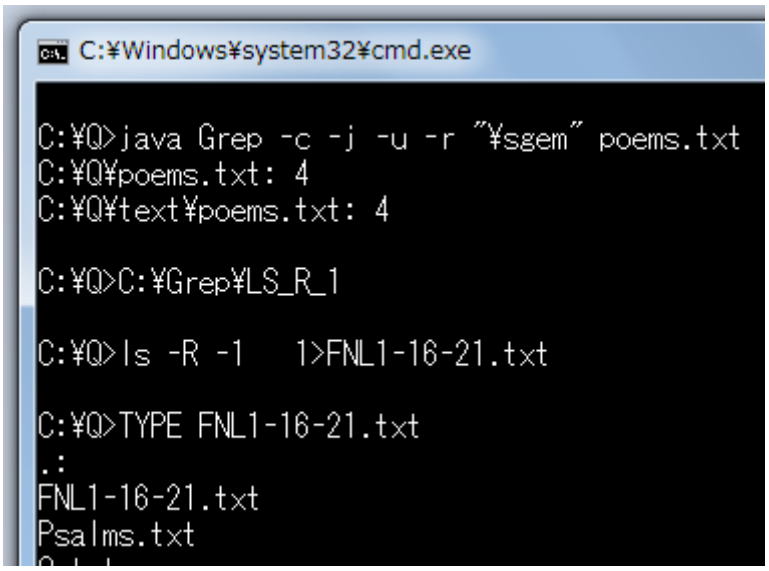
Windows 7 の場合、コマンドプロンプトにバッチファイル名を入力して、Enter を押しますと、バッチファイルに記入した覚えのない 1>が表示されますが、気にしないことにします。標準出力は、1 であり、標準入力は、0 です。Numbers4_sales.txt は、2018 年 8 月のナンバーズ4の売上です。月火水木金にナンバーズ4の抽せんが行われます。月曜日および金曜日は、売上が増加することがわかります。



シェルのワイルドカード表現	正規表現	コマンドプロンプト
.	^.*$	*.*
*	^.*$	*.*
[pqr]oems.txt	^[pqr]oems¥.txt$	ワイルドカードではない。
[!lmn]oems.txt	^[^lmn]oems¥.txt$	ワイルドカードではない。
”^”	^¥^$	”^”
?????.$$$	^.....¥.¥$¥$¥$$	?????.$$$
?	^.$	?
.	^.*¥..*$	*.*
oems.tx	^.*oems¥.tx.*$	*oems.tx*
poem?.txt	^poem.¥.txt$	poem?.txt
poe*.txt	^poe.*¥.txt$	poe*.txt

fileNameListText

検索対象ファイルのリストの名前は、fileName.txtまたはFNL番号−分−秒.txtであり、ソースコードでは、fileNameListTextに記憶されています。−jを付加した例



0 ではない場合のみ

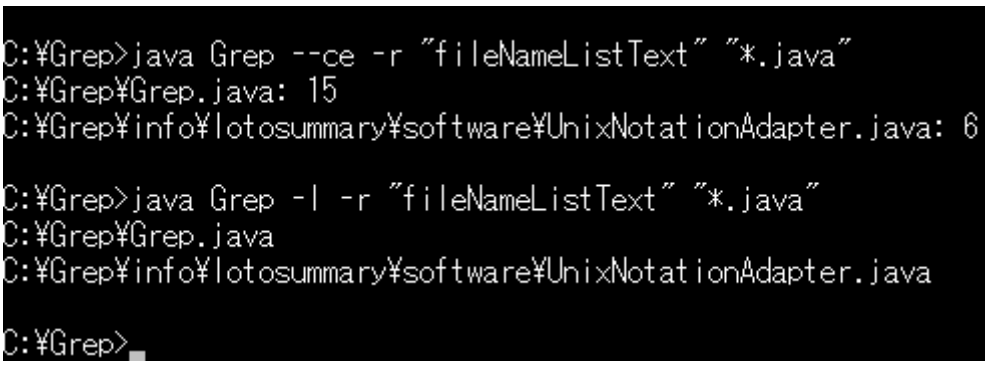
正規表現にマッチする行をカウントして、合計が 0 ではない場合のみファイル名および行数を出力するには、Grep のパラメータとして、--ce を付加してください。

```
java Grep --ce -r "fileNameListText" "*.java"
```

カウントしない

UNIX の grep コマンドは、--ce がありません。正規表現にマッチする行を発見したら、ファイル名を出力して、カウントしないでファイルを閉じて、次のファイルに着手するには、パラメータとして、-l(マイナスエル)を付加してください。-u が指定されていない場合、Java 版の Grep は、ls コマンドではなく、DIR コマンドを利用します。シェルのワイルドカードではなく、コマンドプロンプトのワイルドカードを入力してください。

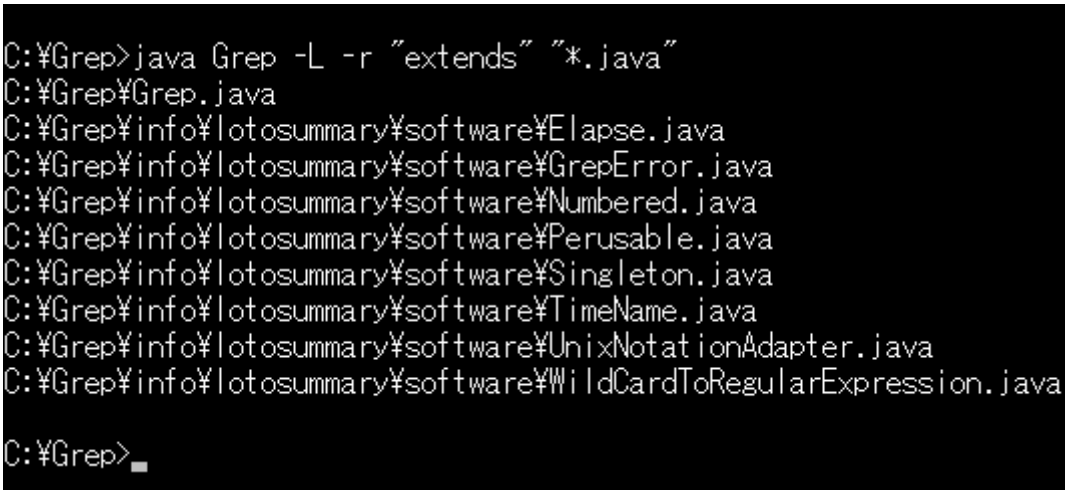
```
java Grep -l -r "fileNameListText" "*.java"
```



カウントされなかったファイル

-Lが指定された場合、Java版のGrepは、正規表現にマッチする行を**発見したら、ファイル名を出力しないで、**カウントして、ファイルを閉じて、次のファイルに着手します。1行もカウントされなかったファイルの名前を出力するには、パラメータとして、-Lを付加してください。-Lでファイル名が表示された場合、Java版のGrepは、Windowsに 1 を返している場合があります。

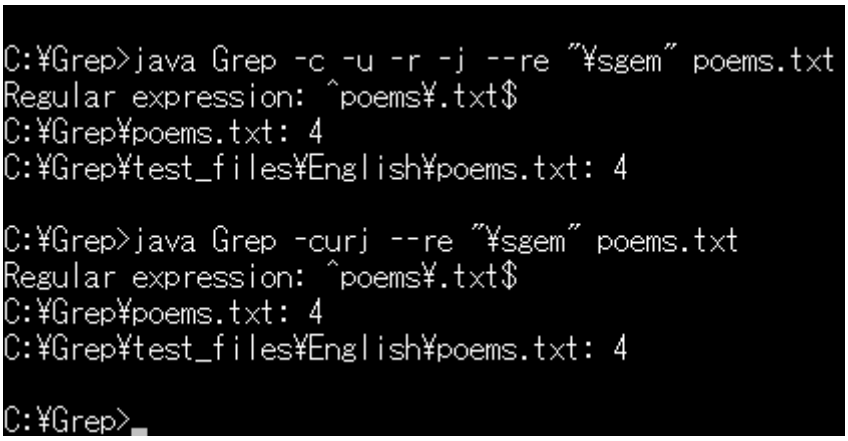
```
java Grep -L -r "extends" "*.java"
```



多数のハイフン＋文字

パラメータとして、多数の**ハイフン＋文字**を入力するのではなく、利用者は、1 個のハイフンに続けて文字列を入力できます。この場合、Java 版の Grep は、文字列を多数の文字に分解して、ハイフンを各文字に前置します。たとえば、-curj は、下記のパラメータを意味します。

```
-c -u -r -j
```



分解されないパラメータ

DIRコマンドを実行しないで、既存の fileName.txt を利用するために、-b-は、分解されません。正規表現追加モードを終了するために、-e-は、分解されません。--ce, --re, --label=など、2 連のハイフンがあるパラメータは、分解されません。ハイフン、英字、数字の場合（-b80, -C4, -m3 など）は、分解されません。漢字などの文字コードを統一するために、-Ke, -Ks, -Ku, -Kw,は、いずれも分解されません。

ファイルが発見されない

-b-が指定されていない場合、Java 版の Grep は、まず、fileName.txt を削除します。、コマンドプロンプトでは、標準出力がリダイレクトされるために、新しい fileName.txt が作成されます。新しい fileName.txt のサイズは、0 です。次に、Java 版の Grep は、DIR コマンドでファイルを検索しますが、ファイルが発見されない場合、サイズが 0 のままで新しい fileName.txt が放置されます。

4 秒が経過

DIR コマンドでファイルを発見できないで、4 秒が経過したとき、Java 版の Grep は、標準エラー出力にエラーメッセージを出力します。DIR コマンドが標準エラー出力に出力するエラーメッセージは、fileName.txt に保存されません。下記のエラーメッセージは、fileName.txt の長さが短すぎることを意味します。ソースコードに記載のとおり、TOO_SMALL の値は、1 であるため、**fileName.txt のサイズは、0 のままである**ことがわかります。

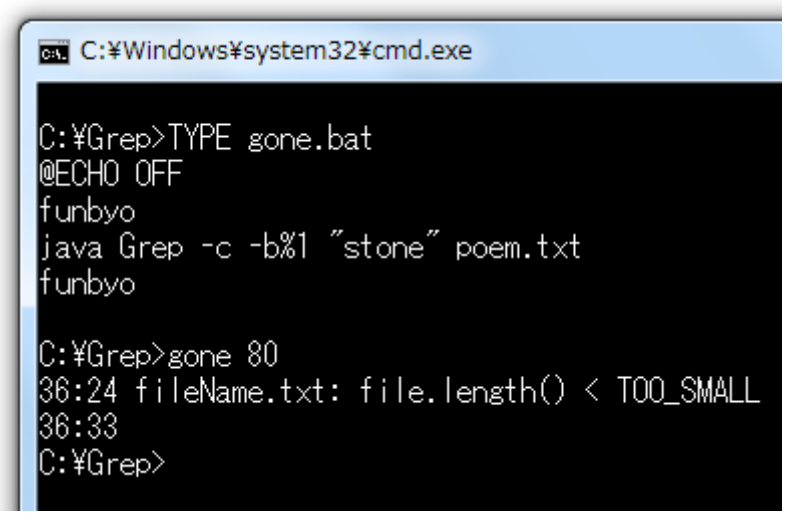
file.length() < TOO_SMALL

待機時間を調節できる

Java 版の Grep が DIR コマンドの結果を待機する時間を調節できます。100 ミリ秒の単位で設定できます。たとえば、4 秒ではなく 8 秒が経過するまで待機させるには、パラメータとして、-b80 を付加してください。b の直後に空白を挿入しないでください。

テストファイルの例

funbyo コマンドは、現在の時刻の分、秒を表示します。funbyo コマンドは、改行しないで終了します。gone.batというテストファイルを添付しました。poems.txtではなくpoem.txtをDIRコマンドにわたしているため、fileName.txtのサイズは、0 のままです。



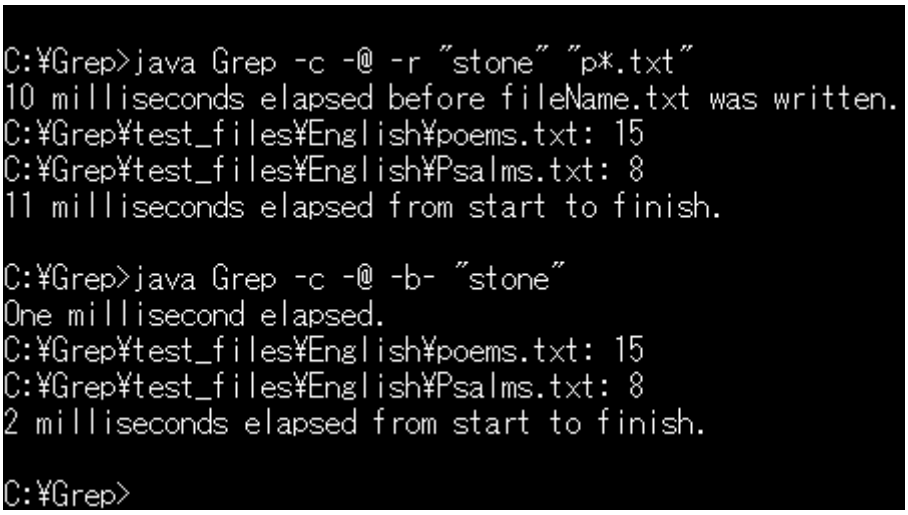
経過時間

Java 版の Grep は、DIR コマンドで fileName.txt を作成する工程と、正規表現にマッチする行を各ファイルから抽出する工程とからなります。fileName.txt を作成する時間および実行の開始から終了まで経過した時間を表示するには、パラメータとして--elapse または-@を付加してください。1 ミリ秒の単位で表示されます。fileName.txt を作成する時間は、経過時間に含まれます。1 ミリ秒未満を測定できないため、Java 版の Grep は、測定が不正確です。

```
java Grep -c -@ "Loyalty" poems.txt
```

-b-で時間が短縮される

-b-が指定された場合、Java 版の Grep は、既存の fileName.txt を読み込みます。この場合、Java 版の Grep は、DIR コマンドを実行しないため、経過時間は、短縮されます。**-b-が指定された場合、-r は、無効です。**



ASCII

バイナリファイルであるかどうか、文字コードがシフト JIS であるかどうか意識しないで、Java 版の Grep を実行するには、パラメータとして、--ascii または-y を付加してください。

java Grep -c -y "99 00" rannu.txt

各行に文字コードを記入しない

--ascii または-y が指定された場合、Java 版の Grep は、検索対象ファイルのリスト（fileName.txt）の各行をファイル名のみであるとみなします。この場合、Java 版の Grep は、バイナリファイルであるかどうか、文字コードがシフト JIS であるかどうか判定しないため、利用者は、各行に文字コードを記入しないでください。

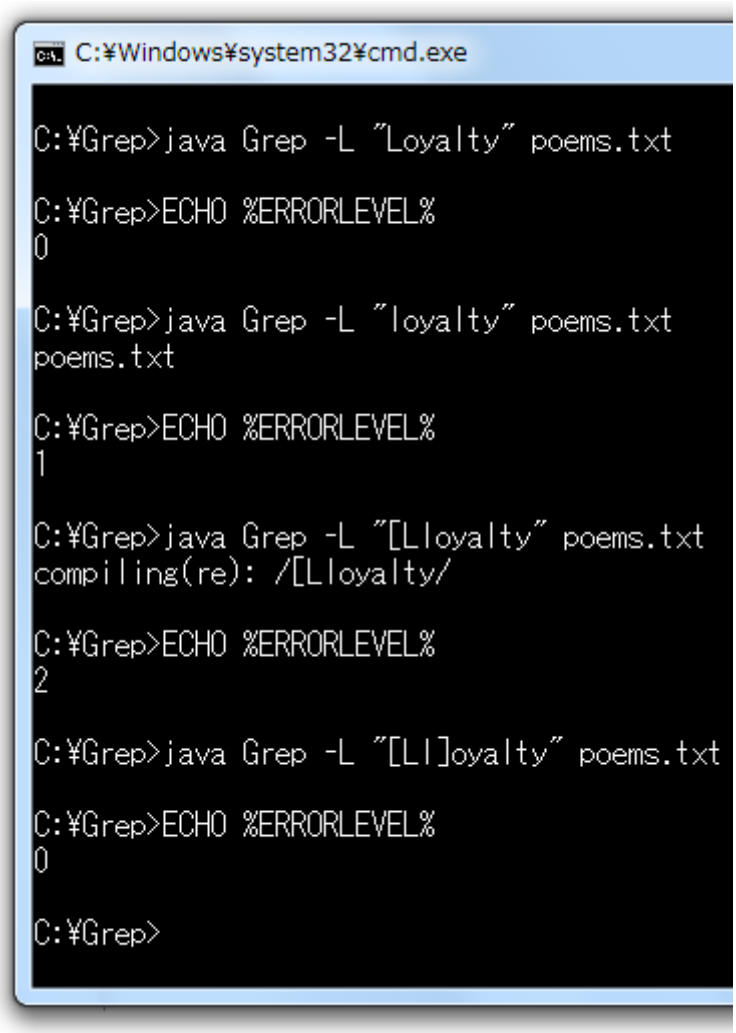
返り値

少なくとも 1 行が正規表現にマッチした場合、Java 版の Grep は、0 を返します。正規表現に 1 行もマッチしなかった場合、Java 版の Grep は、1 を返します。エラーの場合、Java 版の Grep は、2 を返します。

ECHO コマンド

Java 版の Grep が終了したとき、Windows に返した値（返り値、戻り値、リターンコード、終了コード）を表示するには、コマンドプロンプトに、下記のコマンドおよびパラメータを入力して、Enter を押してください。

ECHO %ERRORLEVEL%



エラーメッセージを表示しない

-q を指定した場合、終了するとき、エラーメッセージを表示しません。この場合、Java 版の Grep は、2 ではなく、0 または 1 を返します。【参考】標準エラー出力からファイルに--help による出力をリダイレクトするには、-q を付加してください。

```
java Grep -L -q "[Lloyalty" poems.txt
```

```
C:¥Grep>java Grep -L -q "[Lloyalty" poems.txt

C:¥Grep>ECHO %ERRORLEVEL%
1

C:¥Grep>java Grep -L -q "[Ll]oyalty" poems.txt

C:¥Grep>ECHO %ERRORLEVEL%
0

C:¥Grep>_
```

何も出力しない

何行かマッチしたにもかかわらず、Java 版の Grep が何も出力しないで 0 を返すように命令するには、-q を指定して、> NUL を付加してください。

```
C:¥Grep>java Grep -L "[Lloyalty" poems.txt > NUL
compiling(re): /[Lloyalty/

C:¥Grep>ECHO %ERRORLEVEL%
2

C:¥Grep>java Grep -L -q "[Lloyalty" poems.txt > NUL

C:¥Grep>ECHO %ERRORLEVEL%
1

C:¥Grep>java Grep -L -q "[Ll]oyalty" poems.txt > NUL

C:¥Grep>ECHO %ERRORLEVEL%
0

C:¥Grep>_
```

行の全体

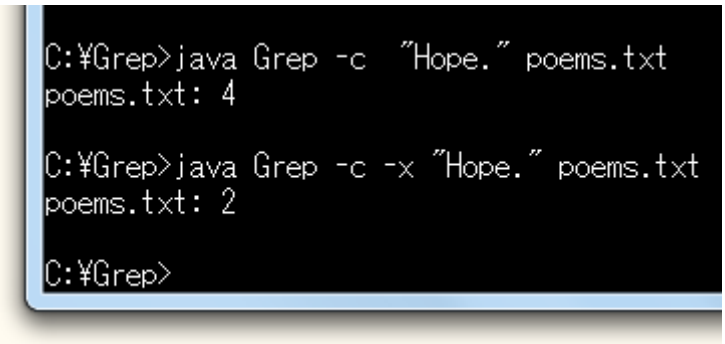
行の全体が正規表現にマッチする場合のみ、ファイルから行を出力するには、パラメータとして、-x を付加してください。

```
java Grep -c -x "Hope." poems.txt
```

カレットおよびドル

-x が指定された場合、Java 版の Grep は、自動的に正規表現の直前にカレット(^)を

付加して、正規表現の直後にドル($)を付加します。^は、行頭を意味します。$は、行末を意味します。-x が指定されたかどうかによって、カウントした結果が異なる例を下図に示します。



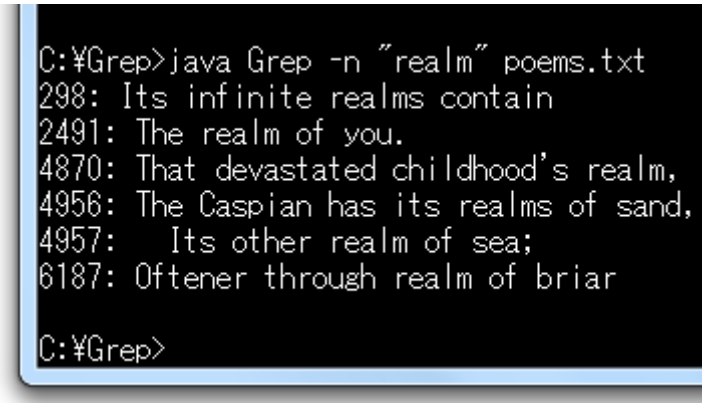
マッチした行を出力

Java 版の Grep は、正規表現にマッチする行をファイルから抽出します。

行番号も必要

行番号、区切り記号、行をこの順に出力するには、パラメータとして、-n を付加してください。

```
java Grep -n "realm" poems.txt
```



ファイル名が必要

ファイル名、区切り記号、行をこの順に出力するには、パラメータとして、-Hを付加してください。ファイル名のみ出力するには、パラメータとして、-l(マイナスエル)を付加してください。

```
java Grep -H -r "regex" "*.java"
```

	カウントする	カウントしない
行を出力する		-H
行を出力しない	--ce	-l

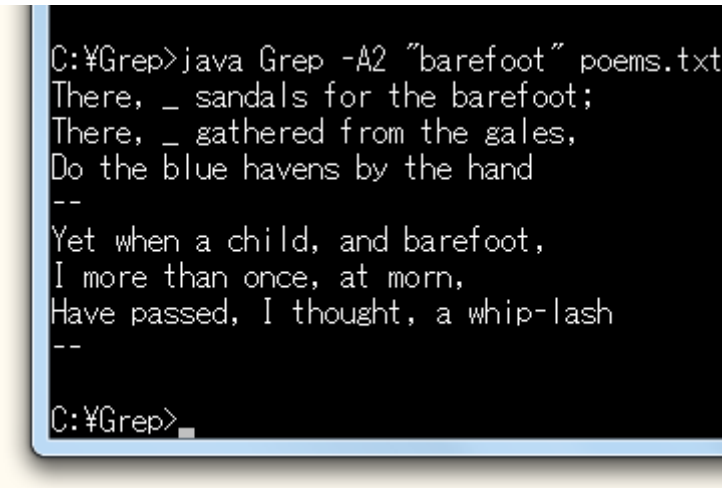
周辺の行を出力

ファイル名も行番号も出力しないで、行のみ出力する場合、たとえば、-A行数、-B行数、-C行数を指定することで周辺の行を出力できます。さもなければ、添付のatndコマンドをお試しください。

-A2

A は、after を意味します。パラメータとして、-A 行数を付加します。A の直後に空白を挿入しないでください。パラメータとして、-A2 が付加された場合、Java 版の Grep は、マッチした行および後続の 2 行を出力します。区切りの行として、ダッシュ(--)が挿入されます。

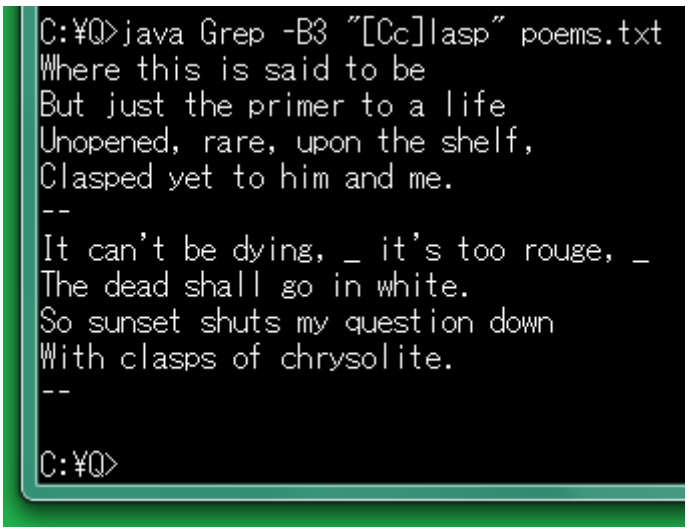
```
java Grep -A2 "barefoot" poems.txt
```



-B3

B は、before を意味します。パラメータとして、-B 行数を付加します。B の直後に空白を挿入しないでください。パラメータとして、-B3 が付加された場合、Java 版の Grep は、マッチした行および先行する 3 行を出力します。区切りの行として、ダッシュ(--)が挿入されます。

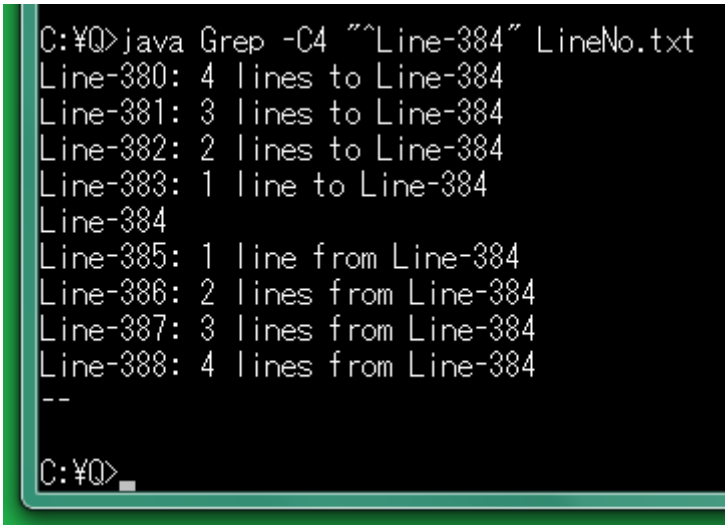
```
java Grep -B3 "[Cc]lasp" poems.txt
```



-C4

マッチした行、先行する 4 行、後続の 4 行を出力します。C は、大文字です。-C4 は、-A4 -B4 と同等です。

java Grep -C4 "^Line-384" LineNo.txt



タグジャンプ

ファイル名、行番号、文字列がこの順で各行を構成しているファイルをサクラエディタで開いて、上向き矢印(↑)および下向き矢印(↓)を押して、カーソルを移動して、F12 を押してください。サクラエディタは、自動的に、そのファイルを開いて、その番号の行にカーソルを移動します。こうしたカーソル移動をタグジャンプと言います。

多機能のテキストエディタ

grep の機能を有しているテキストエディタは、タグジャンプが可能であると思われます。

grep の機能を選択して検索したとき、各行にファイル名、行番号、文字列があるウィ

ンドウが表示されます。MIFES でタグジャンプを行うには、F12 ではなく、F10 を押します。ファイル名、行番号、文字列の区切り文字については、各テキストエディタのヘルプを参照してください。

区切り文字を変更

UNIX の grep は、区切り文字としてコロン(:)を使用します。Windows では、C ドライブを C:として表現するため、Java 版の Grep は、区切り文字として、単なるコロンではなく、コロンおよび空白を使用します。

リダイレクト

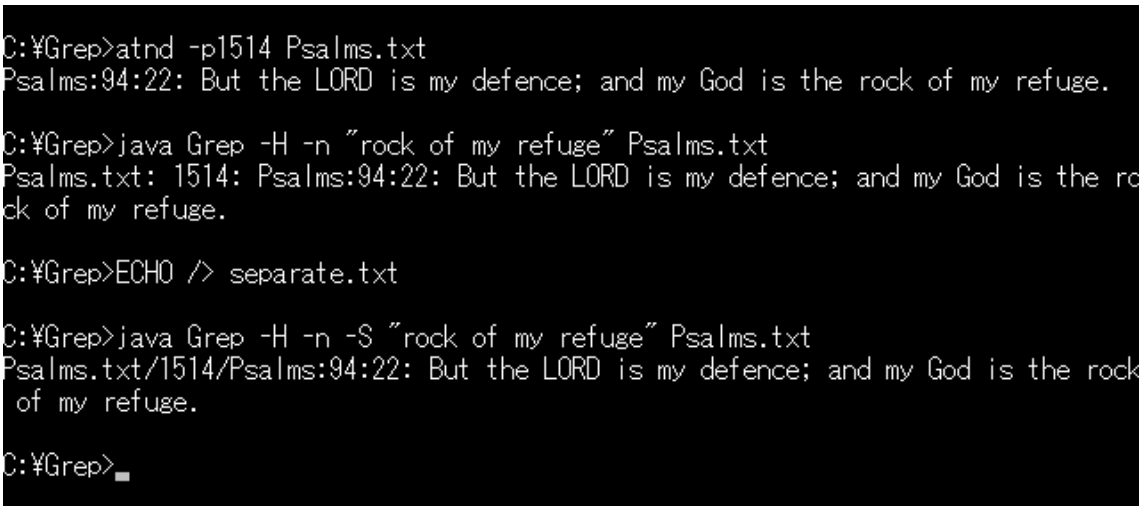
ECHOコマンドは、コマンドプロンプトに文字列、数値を表示します。ECHOコマンドで文字をファイルにリダイレクトすることで、テキストファイルを作成できます。コマンドプロンプトに下記のコマンドおよびパラメータを入力した結果として、テキストファイルは、斜線(/)および改行からなります。

```
ECHO /> separate.txt
```

区切り文字テキスト

Java 版の Grep が区切り文字テキスト(separate.txt)から区切り文字を読み込んで、コロンおよび空白ではなく、新しい区切り文字を使用するように命令するには、パラメータとして、-S を付加してください。S は、大文字です。/に変更した例を図示します。

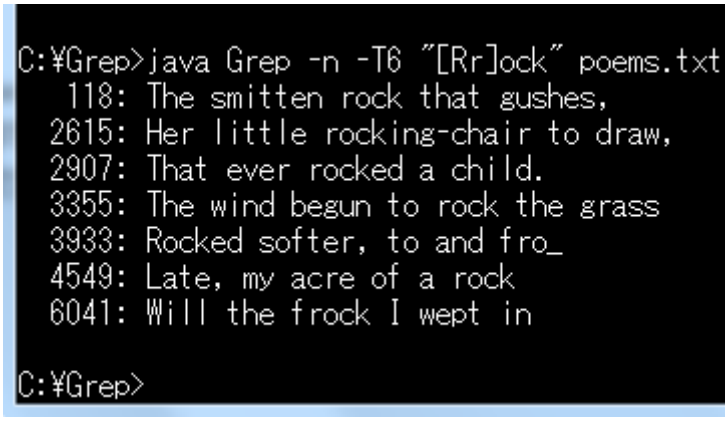
```
java Grep -H -n -S "rock of my refuge" Psalms.txt
```



一の位を固定

行番号が小さくても区切り文字が左にずれないように、行番号の一の位を固定するには、画面の第何桁に行番号の一の位が表示されるか、パラメータとして、-T 桁を付加してください。

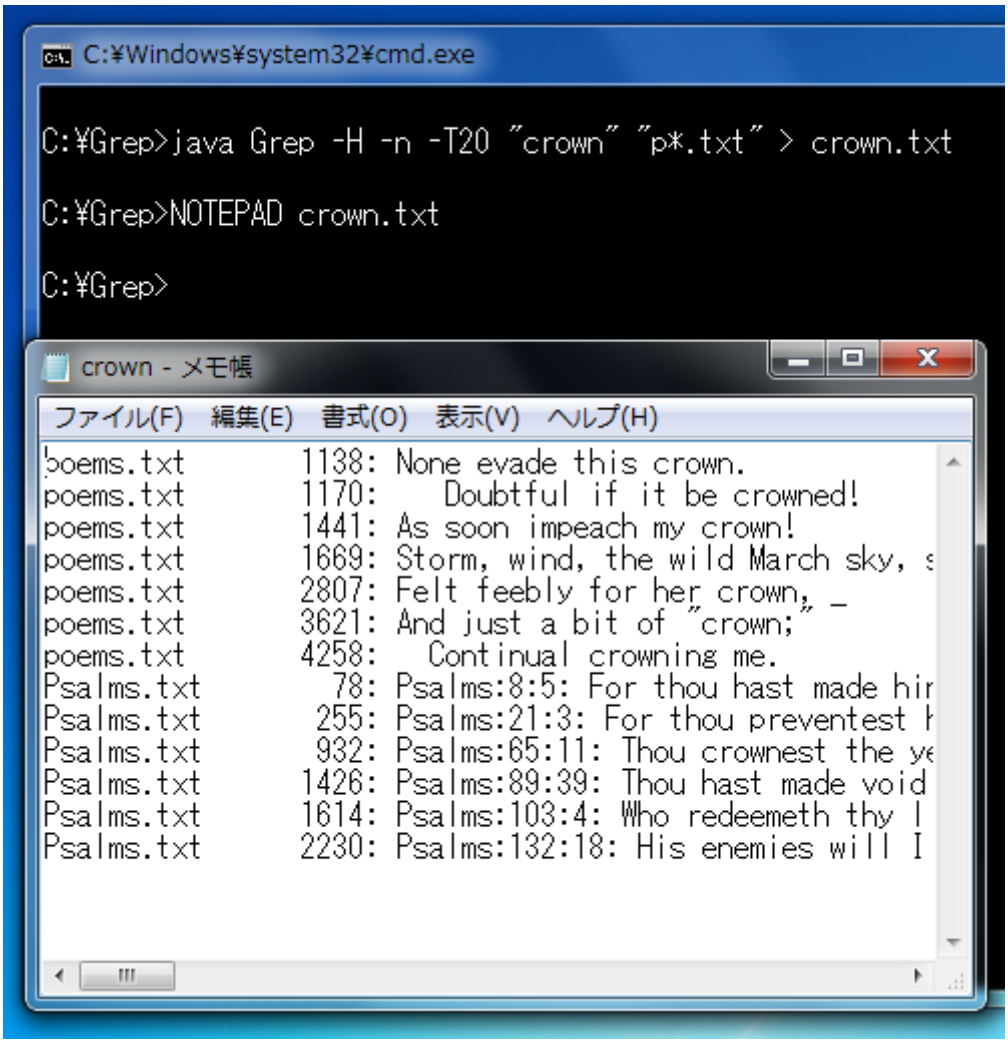
java Grep -n -T6 "[Rr]ock" poems.txt



広い空白

ファイル名および行番号の両方が表示される場合、しかも、行番号の一の位が固定される場合、ファイル名および行番号の間に広い空白が用意されます。

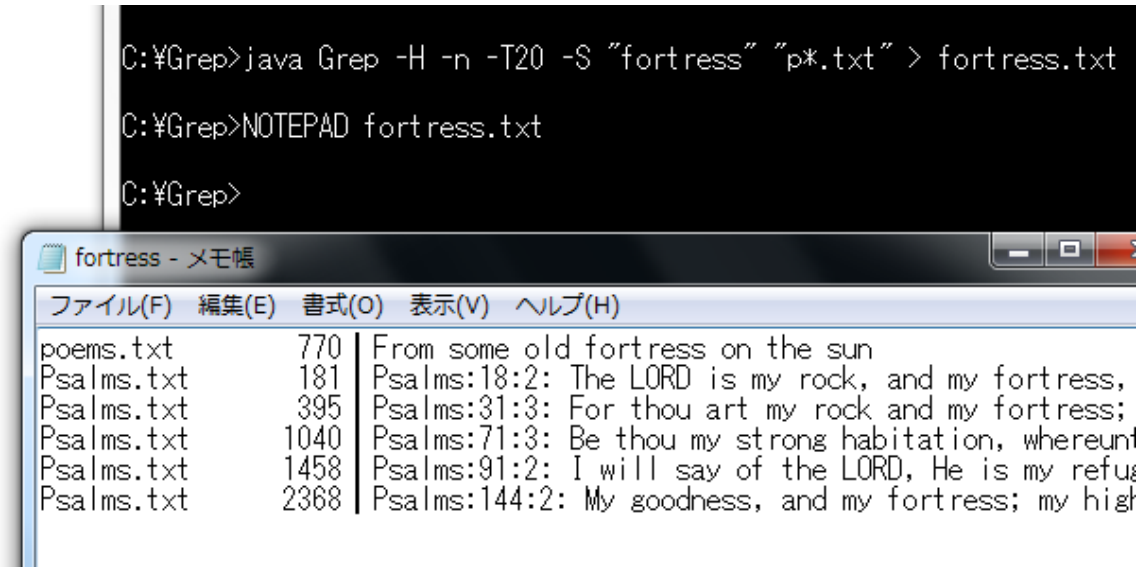
java Grep -H -n -T20 "crown" "p*.txt"



罫線

┼┌└├|などの罫線素片が区切り文字テキスト(separate.txt)に記入された場合、Java 版の Grep は、区切り記号として読み込むことができます。区切り文字テキストに|を記入して、Java 版の Grep に-S で読み込ませた例

```
java Grep -H -n -T20 -S "fortress" "p*.txt"
```



マッチしない行を出力

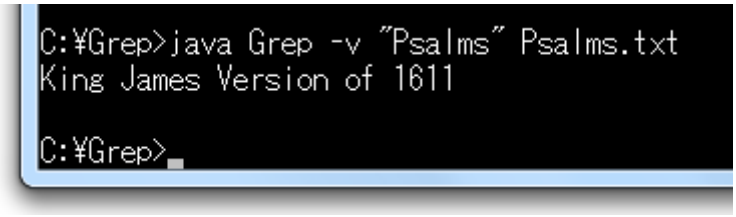
マッチしない行を出力するには、Java 版の Grep のパラメータとして、-v を付加してください。v は、小文字です。

```
java Grep -v "Psalms" Psalms.txt
```

排他的論理和

行が正規表現にマッチしないと判定するために、Java 版の Grep は、排他的論理和を利用します。

パラメータ	-v	0	マッチしなかった	排他的論理和	説明
		1	マッチした		
	0	0		0	
	0	1		1	行を出力する
-v	1	0		1	行を出力する
-v	1	1		0	

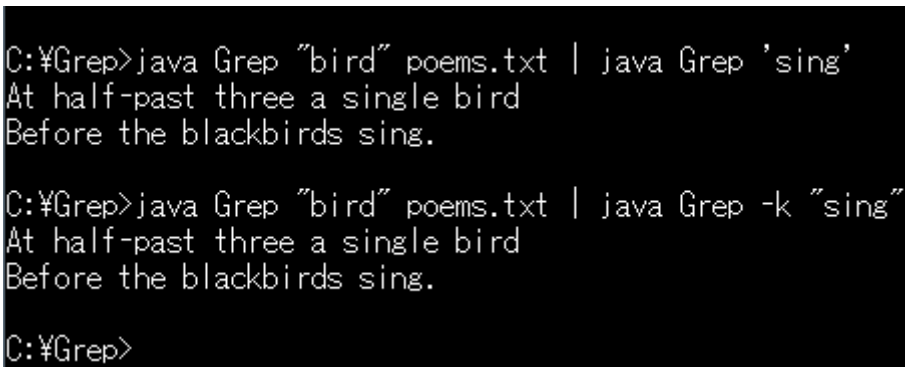


フィルター

フィルターとは、標準入力から入力して、標準出力に出力するコマンドです。多数のフィルターでパイプを構成できます。grepは、正規表現にマッチした行を出力します。grepは、フィルターです。

シングルクォーツ

検索対象のファイルが入力されなかった場合、Java版のGrepが標準入力を情報源とみなすように、引用符(″)ではなく、シングルクォーツで正規表現を囲んでください。さもなければ、パラメータとして-kを付加してください。kは、小文字です。AND検索の例



中止したい合図

コマンドプロンプトに、プログラムの実行を中止したい合図を入力するには、Ctrl を押しながら C を押してください。Ctrl + C は、この操作を意味します。

ウィンドウのサイズ

Windows7 の場合、画面バッファーのサイズは、300 行であり、ウィンドウのサイズとして、高さは、25 行です。Windows 10 の場合、画面バッファーのサイズは、9001 行であり、ウィンドウのサイズとして、高さは、30 行です。コマンドプロンプトのウィンドウのプロパティを表示するには、コマンドプロンプトのタイトルバーを右クリックして、[レイアウト]タブをクリックしてください。ウィンドウのプロパティで、利用者は、画面バッファーおよびウィンドウのサイズを変更できます。

上限

正規表現にマッチした行が多すぎるかもしれません。たとえば、3 行を出力したら、ファイルを閉じて、次のファイルに着手するには、パラメータとして、-m3 を指定してください。m の直後に空白を挿入しないでください。

```
java Grep -m3 ″bird″ ″p*.txt″
```

```
C:¥Grep>java Grep -m3 "bird" "p*.txt"
bird
And blushing birds go down to drink,
And birds of foreign tongue!
The birds jocoser sung;
Psalms:11:1: In the LORD put I my trust: How say ye to my soul, Flee as a bird t
o your mountain?
Psalms:104:17: Where the birds make their nests: as for the stork, the fir trees
 are her house.
Psalms:124:7: Our soul is escaped as a bird out of the snare of the fowlers: the
 snare is broken, and we are escaped.

C:¥Grep>
```

文字コード

Windows のコマンドプロンプトは、シフト JIS が使用される場合が多い。Java は、Unicode の文字列をサポートします。Windows のコマンドプロンプトで日本語をサポートする grep が必要である場合、Java 版の Grep を試すことができます。

CHCP コマンド

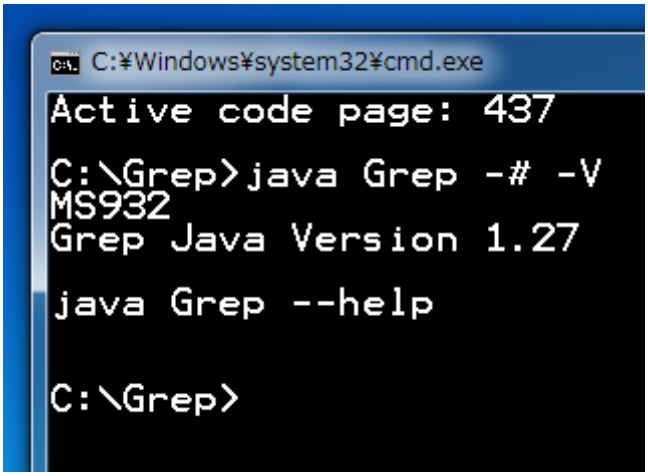
規格として、シフトJISはなく 932 のコードページ(MS932)がWindowsのコマンドプロンプトに使用されます。MS932 は、いわゆるシフトJISですが規格外です。MS932 は、Windows-31Jとも言います。932を表示するには、コマンドプロンプトに、CHCPコマンドを入力して、パラメータを入力しないで、Enterを押してください。Java版のGrepのパラメータとして、-#を付加することで、Java環境変数(file.encoding)の値を出力できます。標準出力をリダイレクトしたファイルの文字コードは、MS932 であることがわかります。バージョン番号を表示する-Vは、-#と併用できます。

```
java Grep -# "a" bell.txt
```

高速化の課題

英語(アメリカ)のコードページは、437 または 1252 です。--ascii も-y も、利用者が文字コードを意識しない素朴なモードに切り替えます。入力および出力を高速化するに

は、コマンドプロンプトのコードページを日本語ではなく英語にするべきであると思われますが、コードページを 437 に変更しても、Java 環境変数の値は、MS932 であることがわかります。



各行に文字コード

grep と異なり、Java 版の Grep は、検索対象ファイルのリスト（fileName.txt）の各行から文字コードを捕捉できます。事前に、利用者は、検索対象のファイルのリスト（fileName.txt）の各ファイル名に文字コードを記入できます。記入済みの fileName.txt を利用するために、パラメータとして、-b-を付加してください。

小なり記号

記入の形式として、ファイル名および文字コードの間を小なり記号で区切ってください。空白を含めないでください。

Samuel.txt<UTF-8

判定しない

パラメータとして、**--ascii も-y も文字コードが ASCII であることを指定するのではありません。**--ascii または-y が指定された場合、Java 版の Grep は、検索対象ファイルのリスト（fileName.txt）の各行で点検しないで、どの行もファイル名であるとみなします。すなわち、Java 版の Grep は、バイナリファイルであるかどうか、文字コードがシフト JIS であるかどうか判定しないため、利用者は、fileName.txt の各行に文字コードを記入しないでください。

文字コードを統一

Java 版の Grep は、fileName.txt の各行から文字コードを捕捉できますが、どのファイルも同じ文字コードである場合、fileName.txt の各行に文字コードを記入しないで、パラメータとして、--encoding=文字コードを付加してください。

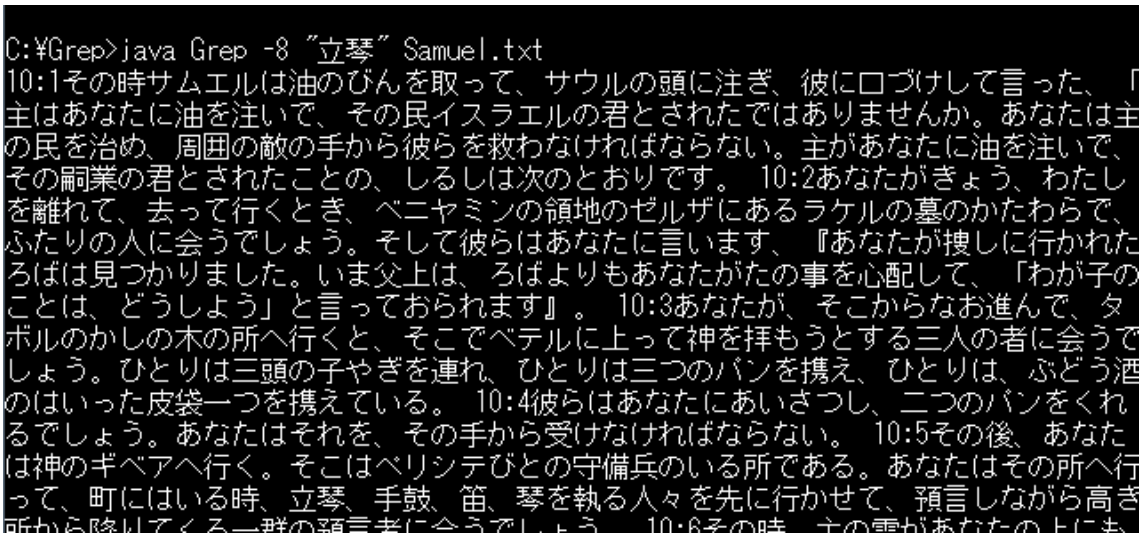
コマンドプロンプトで日本語 IME

コマンドプロンプトに日本語を入力するには、まず、Alt を押しながら、[半角／全角]キーを押してください。Windows 7 の場合、Alt を押す必要がありません。次に、かな漢字変換を行ってください。かな漢字変換モードから脱出するには、[半角／全角]キーを押してください。

同等なパラメータ

fileName.txtの各行に文字コードを記入しないで、漢字などの文字コードを統一するパラメータとして、-Ke, -Ks, -Ku, -Kw,は、それぞれ、EUC-JP, Shift_JIS, UTF-8, MS932を意味します。-8 は、-Kuと同じ結果を生じます。1 個のハイフンに続けて文字列を入力できます。この場合、Java版のGrepは、文字列を多数の文字に分解して、ハイフンを各文字に前置します。ただし、--encoding=UTF-8 も-Kuも分解されません。-8 は、他のパラメータと組み合わせて、1 個のハイフンに続けることができます。ただし、組合せの 2 文字目に 8 を入力しないでください。たとえば、-c8 ではなく、-8cにしてください。-8cは、-8 -cと同等です。

java Grep -8 "立琴" Samuel.txt



コードページ

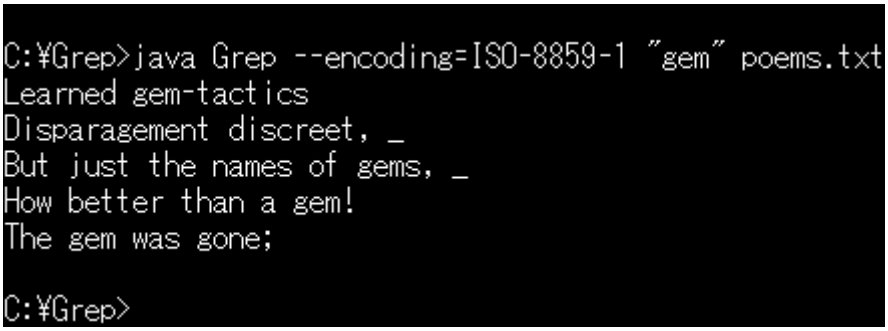
日本語のコードページは、932 ですが、英語（アメリカ）のコードページは、437 または 1252 です。日本語版 Windows の文字コードは、Windows-31J ですが、英語版 Windows の文字コードは、Windows-1252 です。パラメータとして、--encoding=CP1252 を付加できます。

ISO-8859-1

英語であることをGrepがJavaに伝達するように命令するには、パラメータとして、--encoding=ISO-8859-1 を付加してください。=の直後に空白を挿入しないでください。このパラメータにより、Java版のGrepは、ラテン文字-1 のファイルを入力しますが、

MS932 の行を出力します。高速化の観点で、入力のみ改善されると思われます。

java Grep --encoding=ISO-8859-1 "gem" poems.txt



Java が処理する文字コード	パラメータ	説明
--encoding=ASCII		
--encoding=Cp1252		Windows ラテン文字-1
--encoding=EUC-JP	-Ke	UNIX の日本語
--encoding=EUC_JP	-Ke	UNIX の日本語
--encoding=ISO-8859-1		ラテン文字-1
--encoding=ISO8859_1		ラテン文字-1
--encoding=MS932	-Kw	Windows 日本語、Windows-31J
--encoding=Shift_JIS	-Ks	
--encoding=SJIS	-Ks	
--encoding=UTF-16		
--encoding=UTF-8	-Ku	
--encoding=UTF8	-Ku	

sjisUTF8 コマンド

トリシーカーから sjisUTF8-C.zip および sjisUTF8.zip をダウンロードできます。それぞれ、C 言語版および C++版です。ただし、フィルターとして sjisUTF8 コマンドを利用できません。sjisUTF8 コマンドは、シフト JIS のファイルを読み込んで、UTF-8 のファイルを書き出します。改行するとき、CRLF ではなく LF を使用できます。

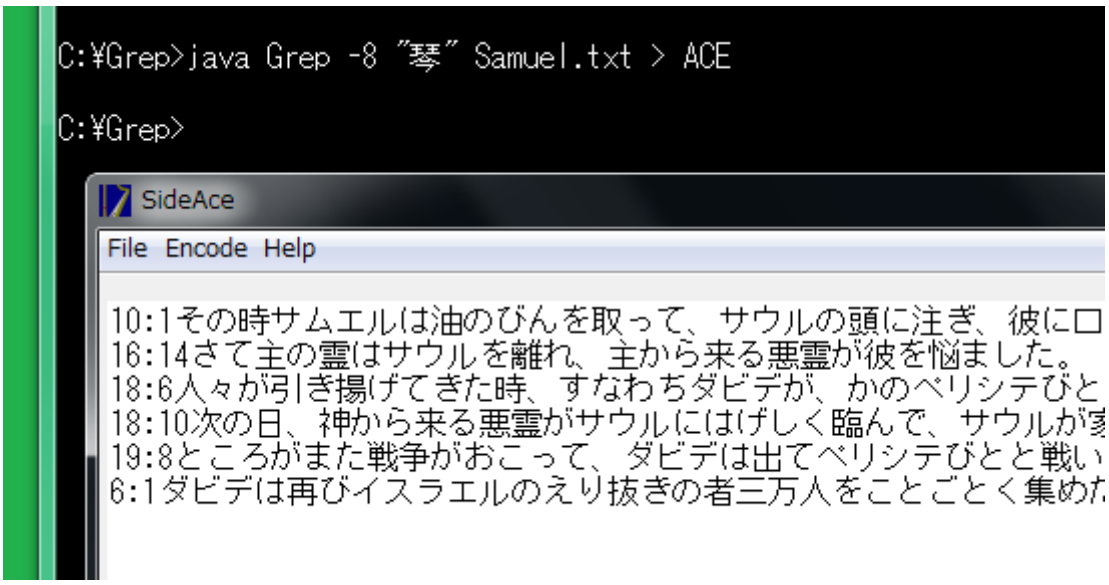
SideAce ウィンドウ

Java 版の Grep に Swing 版の SideAce を添付しました。これは、SideAce ウィンドウを表示します。SideAce は、ACE というファイルを作成します。ACE は、Windows のファイル名であるため、大文字と小文字が区別されません。SideAce は、ACE を監視しています。ACE が書き込まれた時刻が更新されたら、MS932 のファイル（ACE）の全部を読み込んで、SideAce ウィンドウに冒頭を表示します。利用者が SideAce ウィンドウを閉じたとき、**ACE は、削除されます。**

ACE にリダイレクト

Java 版の Grep の出力を ACE にリダイレクトしたとき、SideAce ウィンドウに出力が表示されます。コマンドプロンプトは、いわゆる**折り返し**を行います。この場合、論理的な 1 行は、多数の行として表示される場合があります。コマンドプロンプトと異なり、SideAce ウィンドウは、論理的な 1 行を 1 行で表示します。この場合、長すぎる行は、左側にある一部しか表示されません。

java Grep -8 "琴" Samuel.txt > ACE



連携

Swing 版の SideAce と連携するには、Java 版の Grep のパラメータとして、--ace を付加してください。ace は、小文字です。START_ACE.bat というバッチファイルが実行されて、コマンドプロンプトのウィンドウとは独立した SideAce ウィンドウが表示されます。すでに、SideAce ウィンドウが表示されている場合、--ace は、不要です。

java Grep --ace -8 "琴" Samuel.txt > ACE

ウィンドウから書き出す

ACE は、MS932 のファイルでしたが、UTF-8 のファイルとして ACE を保存するには、SideAce ウィンドウの File メニューから Leave を選択してください。SideAce ウィンドウは、閉じられます。

Encode メニュー

UTF-8 ではなく EUC-JP のファイルとして ACE を保存するには、まず、Encode メニューから EUC-JP を選択してください。次に、File メニューから Leave を選択してください。SideAce ウィンドウは、閉じられます。

消去したい

SideAce は、ACE を読み込んで、あらゆる行を記憶しています。ACE の冒頭の左側にある一部しか SideAce ウィンドウに表示されていません。SideAce ウィンドウに表示されたテキストを消去するには、ステータスバーをクリックしてください。1 個の空白のみ表示されます。ただし、記憶は、消去されていません。File メニューから Leave を選択することで、全部を保存できます。

カレントディレクトリ

SideAce ウィンドウの下部にあるステータスバーには、Java が認識しているカレントディレクトリ(user.dir)が表示されています。Encode メニューから EUC-JP を選択したとき、ステータスバーに EUC-JP が表示されます。SideAce が ACE を読み込んだとき、カレントディレクトリがステータスバーに表示されます。

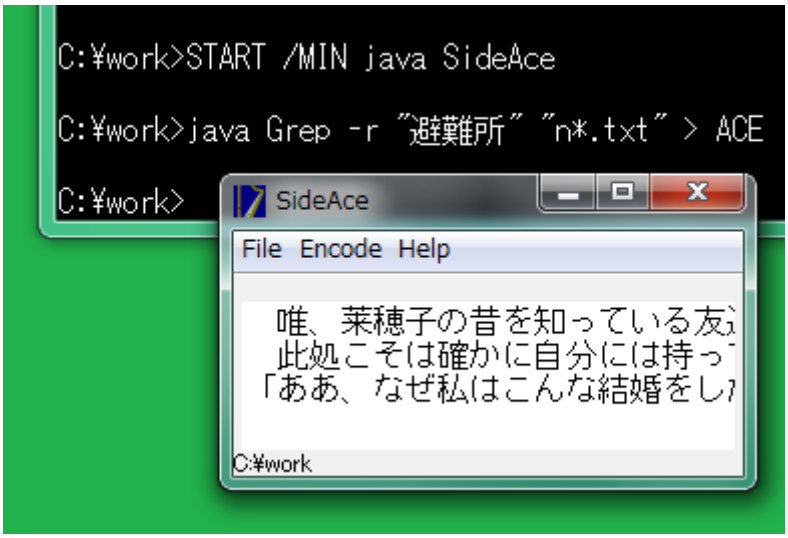
起動できるか

ACE のサイズが 0 より大きい場合は、SideAce を起動できません。ACE の名前を変更してください。さもなければ、**ACE を削除してください。**ACE が発見されない場合は、SideAce を起動できます。コマンドプロンプトは、> ACE の入力により、サイズが 0 である ACE を作成しますが、この場合は、SideAce を起動できます。

バッチファイル

START_ACE.bat というバッチファイルを実行することで、SideAce ウィンドウを表示できます。下線(_)を入力するには、Shift を押しながら、ろを押してください。START_ACE.bat を削除しないでください。START_ACE.bat が発見されない場合、Java 版の Grep は、自動的に C:¥Grep¥START_ACE.bat を生成します。内容は、以下のとおりです。

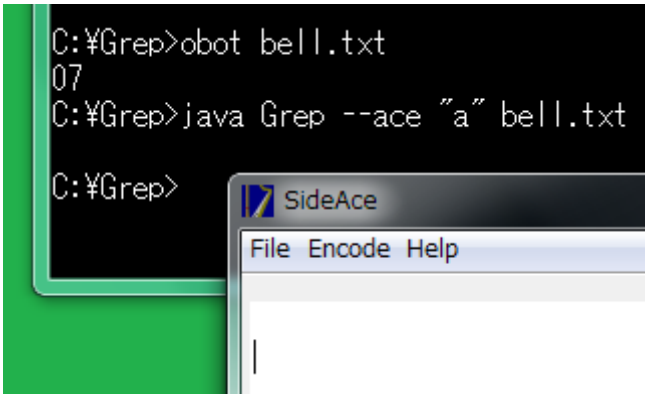
```
START /MIN java SideAce
```



bell.txt

bell.txtは、1 バイトのベル文字（07h）からなるファイルです。obotコマンドで確認できます。aを検索しても失敗しますが、パラメータとして、--aceを付加することで、SideAceを起動できます。

```
java Grep --ace "a" bell.txt
```



もうひとつのコマンドプロンプト

もうひとつのコマンドプロンプトを表示するには、Windows ロゴキーを押しながら、R を押して、cmd を入力して、Enter を押してください。Java 版の Grep を使用するディレクトリに CD コマンドで移動してください。SideAce を起動するには、コマンドプロンプトに、下記のコマンドおよびパラメータを入力して、Enter を押してください。SideAce の大文字と小文字は、区別されます。

```
java SideAce
```

ダブルクリック

Swing 版の SideAce に doubleClick.exe を添付しました。エクスプローラーで doubleClick というアプリケーションを表示してダブルクリックすることで、SideAce ウィンドウを表示できます。doubleClick は、SideAce ウィンドウからヘルプの文書（このPDF）を閲覧するために必要です。手順の例を示します。フォルダのことをディレクトリとも言います。

1. まず、C:¥Grep¥doubleClick.exe をカレントディレクトリにコピーしてください。
2. いったんコマンドプロンプトを閉じてください。
3. Windows ロゴキー（田キー）を押しながら、E を押してください。エクスプローラーが起動します。
4. ステップ 1 でコピーした doubleClick というアプリケーションをダブルクリックしてください。SideAce ウィンドウが開きます。
5. SideAce ウィンドウの下部にあるステータスバーにカレントディレクトリが表示され

ているのを確認してください。

6. この時点で、コマンドプロンプトを開いてください。
7. CD コマンドで doubleClick のカレントディレクトリに移動してください。

doubleClick コマンド

doubleClick.exe のバージョンおよびコマンドパラメータを表示するには、コマンドプロンプトに、まず下記のコマンドおよびパラメータを入力して、Enter を押してください。次に、version.txt を閲覧してください。

```
doubleClick -VC
```

大きいウィンドウ

SideAce ウィンドウを大きくするには、Help メニューから Large を選択してください。さもなければ、SideAce ウィンドウの右側にある辺縁でポインティングデバイス（マウス、タッチパッドなど）のアイコンが I 形ではなく矢印になった状態でクリックしてください。標準のサイズに戻すには、Help メニューから Standard を選択してください。最大化されたウィンドウと異なり、大きいウィンドウは、タイトルバーをドラッグすることで画面を移動できます。最大化ボタンと異なり、Large ボタンが押されたとき、SideAce は、ウィンドウの幅から表示桁数を算出します。

スクロール

画面バッファーのサイズが大きい場合、コマンドプロンプトのウィンドウは、高さ方向にスクロールバーが付与されます。コマンドプロンプトのカーソルは、下端にありますが、利用者は、スクロールバーで画面の上方を閲覧できます。

MORE コマンド

MOREコマンドのヘルプを参照するには、コマンドプロンプトで、**MORE /?**を実行してください。コマンドの出力の最後から最初へ、スクロールするのではなく、出力された順に閲覧するには、標準出力をMOREコマンドに入力するパイプを実行してください。スペースキーで進行します。Qで終了します。Ctrl + Cの操作で中止できます。

```
java Grep -8 "ダビデ" Samuel.txt | MORE
```

クリップボード

どの Windows も最初からクリップボードを用意しています。コマンドプロンプトのウィンドウとは独立したメモ帳ウィンドウに、クリップボードからコマンドの出力を貼り付けることができます。

CLIP コマンド

Windows Server 2003, Windows Vista, Windows 7, Windows 10 には、CLIP コマンドが

用意されています。CLIP コマンドは、パイプを構成できます。コマンドの出力を貼り付ける手順を示します。

1. Windows のメモ帳で新しいファイルを表示してください。
2. 標準出力を CLIP コマンドに入力するパイプを実行してください。
3. タスクバーからメモ帳を選択してください。
4. Ctrl を押しながら V を押すことで、新しいファイルに貼り付けてください。Ctrl + V は、この操作を意味します。

java Grep -8 "ダビデ" Samuel.txt | CLIP