

AceGrep ユーザーズガイド

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、`acegrep` コマンドを試すことができます。ソースコードを添付しました。

`acegrep` “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

AceGrep 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

`acegrep` コマンドのパラメータとして、`-u` が付加された場合、AceGrep 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。

ファイル名の大文字と小文字を区別

DIR コマンドと異なり、`ls` コマンドは、ファイル名の大文字と小文字を区別します。`ls` コマンドと同様に、AceGrep に内蔵の AceList は、ファイル名の大文字と小文字を区別できます。`acegrep` コマンドのパラメータとして、`--in` が付加された場合、AceGrep は、DIR コマンドも `ls` コマンドも実行しないで、AceList で `fileName.txt` を作成します。

事前に用意

事前に利用者が `fileName.txt` を用意した場合、しかも `acegrep` コマンドのパラメータとして、`-b-` が付加された場合、AceGrep は、DIR コマンドも `ls` コマンドも AceList 機能も実行しないで、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、AceGrep 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せんが、ファイル名の正規表現も使用できます。同様に、AceGrep に内蔵の AceList を利用する場合、UNIX のシェルのワイルドカードも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

AceGrep 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、AceGrep 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、AceGrep 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

AceGrep は、C# のコンソールアプリケーションです。AceGrep は、.NET Framework 3.5 または .NET Framework 4 を利用できます。

手順

1. トリシーカー (<http://tori.tobihiro.jp/>) というウェブサイトから AceGrep.zip をダウンロードしてください。
2. AceGrep.zip を展開してください。
3. たとえば、C:\¥AceGrep¥AceGrep.exe となるように、AceGrep 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、Path を選択して、[編集]ボタンを押してください。Path は、PATH とも書きます。
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. 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。
10. セミコロン(;)および C:¥AceGrep を追加してください。末尾に入力した例

;C:¥AceGrep

11. 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:¥AceGrep)ごと削除してください。Path から C:¥AceGrep を削除してください。

著作権

Sogaya(そがや)は、AceGrep の著作権を保有しています。

免責条項

AceGrep 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。

Visual C# 2008 Express

Visual C# 2008 Express または Visual Studio 2010 で構築されています。.NET Framework 3.5 または.NET Framework 4 を利用できます。

ユーザーズガイド

2018 年 12 月 1 日。Revision 1.012

AceGrep のパラメータ

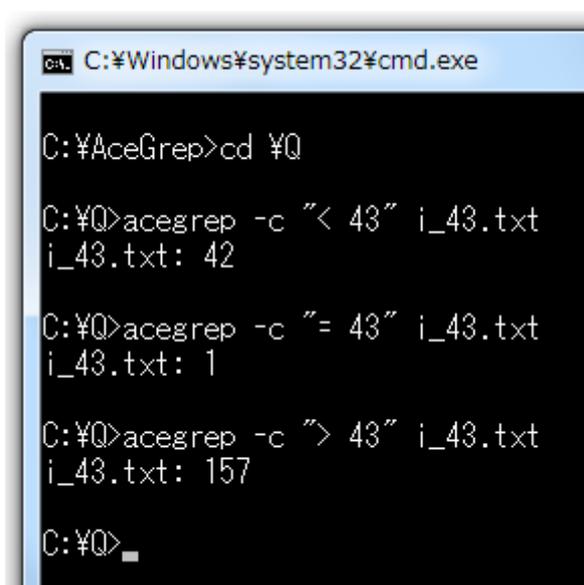
テストファイルを添付しました。英語のテストファイルが、English というディレクトリにあります。日本語のテストファイルが、Japanese というディレクトリにあります。AWK のスクリプトが、test_files というディレクトリにあります。以下の説明で、こうしたテストファイ

ルを利用します。

ファイル名および行数

UNIX の `grep` と同様に、AceGrep 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、`acegrep` コマンド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。

```
acegrep -c "< 43" i_43.txt
```



```
C:\Windows\system32\cmd.exe
C:\AceGrep>cd %Q
C:\AceGrep>acegrep -c "< 43" i_43.txt
i_43.txt: 42
C:\AceGrep>acegrep -c "= 43" i_43.txt
i_43.txt: 1
C:\AceGrep>acegrep -c "> 43" i_43.txt
i_43.txt: 157
C:\AceGrep>_
```

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、`acegrep` コマンド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。

```
acegrep -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、AceGrep 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

acegrep -c -v "¥a" i_43.txt

obot コマンド

AceGrep 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(マイナスイェル)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。

```
C:¥Q>awk -f wc.awk i_43.txt
200 lines

C:¥Q>awk -f wc.awk poems.txt
6247 lines

C:¥Q>awk -f wc.awk poems.txt i_43.txt
6447 lines

C:¥Q>
```

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

```
C:¥Q>od -tx1 bell.txt
00000000 07
00000001

C:¥Q>
```

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

AceGrep 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。

```
C:¥Q>obot bell.txt
07
C:¥Q>acegrep -c "¥x07" bell.txt
bell.txt: 1

C:¥Q>acegrep -c "¥a" bell.txt
bell.txt: 1

C:¥Q>acegrep -c -v "¥a" i_43.txt
i_43.txt: 200

C:¥Q>newlines -c i_43.txt
200 lines

C:¥Q>atnd -LF -p1 i_43.txt
i_43.txt: 200 lines
maximum LF-to-LF span of 14 bytes (line-100)
Line 1 < 43

C:¥Q>
```

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、AceGrep 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。

OR 検索

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。

```
acegrep -c -f patterns.txt poems.txt
```

```
C:¥Q>TYPE Q.txt
[Jj]ewel
[Gg]em[^e]
¥s[Rr]ing
[Ee]arring
[Bb]rooch
[Cc]lasp
[Cc]hain
[Nn]ecklace
[Bb]eads
[Ww]ampum
[Cc]rown
[Cc]oronet
[Dd]iadem
[Ll]aurel

C:¥Q>acegrep -c -f Q.txt poems.txt
poems.txt: 37

C:¥Q>
```

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、AceGrep 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。

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。AceGrep 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。

acegrep -b- -c -f Q.txt

```
C:¥Q>TYPE Q.txt
[Jj]ewel
[Gg]em[^e]
¥s[Rr]ing
[Ee]arring
[Bb]rooch
[Cc]lasp
[Cc]hain
[Nn]ecklace
[Bb]eads
[Ww]ampum
[Cc]rown
[Cc]oronet
[Dd]iadem
[Ll]aurel

C:¥Q>TYPE fileName.txt
poems.txt

C:¥Q>acegrep -b- -c -f Q.txt
poems.txt: 37

C:¥Q>
```

1 行で 2 個以上マッチするか

AceGrep は、少なくとも 1 個一致する行をカウントします。AceGrep 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、AceGrep が OR 検索を行った行数より大きい(15 + 2 > 16)。

```
C:¥Q>TYPE G.txt
[Ee]arring
[Ss]tone

C:¥Q>acegrep -c "[Ss]tone" poems.txt
poems.txt: 15

C:¥Q>acegrep -c "[Ee]arring" poems.txt
poems.txt: 2

C:¥Q>acegrep -c -f G.txt poems.txt
poems.txt: 16

C:¥Q>
```

バージョン番号

AceGrep 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。

acegrep -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ってください。多数の正規表現が追加されます。

```
acegrep -c -e "[Ee]arring" "[Ss]tone" -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。AceGrep 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。AceGrep 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。

```
C:¥Q>acegrep -c -e "[Ee]arring" "[Ss]tone" -e poems.txt
poems.txt: 16

C:¥Q>TYPE patterns.txt
[Ee]arring
[Ss]tone

C:¥Q>
```

-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。

```
acegrep -e "[Ee]arring" "[Ss]tone" -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 --dir - -b- -H -k -Kw -p- -u
--ascii --elapsed -@ -c -i -Ke -L -q -v
--bat --in -[ -f -I -Ks -l -r -x
--ce --re -8 -g -j -Ku -n -S -y
```

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。

```
C:¥Q>acegrep -c "[^s][Ww]ind[^o]" poems.txt
poems.txt: 27

C:¥Q>acegrep -c -i "[^s]wind[^o]" poems.txt
poems.txt: 27

C:¥Q>_
```

一時ファイルを命名する

AceGrep 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

AceGrep 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、AceGrep 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

AceGrep 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
7
C:¥Q>TIME /T
20:34
C:¥Q>funbyo
34:47
C:¥Q>acegrep -c -j "[^dfp][Rr]ank" poems.txt
poems.txt: 6
C:¥Q>DIR FNL*.txt | acegrep 'FNL'
2018/11/25 20:34          11 FNL8-34-47.txt
C:¥Q>
```

rvoy コマンド

AceGrep 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

AceGrep 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 から認識できません。

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、AceGrep 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:¥Q>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:¥Q>acegrep -c "no" sayno.txt
sayno.txt: 2

C:¥Q>acegrep -c 'no' sayno.txt
sayno.txt: 2

C:¥Q>acegrep -c "'no'" sayno.txt
sayno.txt: 1

C:¥Q>_
```

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。AceGrep 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、標準入力の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、AceGrep 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 | acegrep -c 'FNL'

```
C:¥Q>DEL FNL*.txt

C:¥Q>acegrep -c -j "stone" poems.txt
poems.txt: 15

C:¥Q>DIR FNL*.txt | acegrep -c 'FNL'
stdin: 1

C:¥Q>DIR FNL*.txt | acegrep -c -k "FNL"
stdin: 1

C:¥Q>
```

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。AceGrep 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 | acegrep -c 'FNL' - bell.txt

```
C:¥Q>DEL FNL*.txt

C:¥Q>acegrep -c -j "barefoot" poems.txt
poems.txt: 2

C:¥Q>DIR FNL*.txt | acegrep -c 'FNL' - bell.txt
stdin: 1
bell.txt: 0

C:¥Q>_
```

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

例えば、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:¥Q>DEL FNL*.txt

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

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

C:¥Q>
```

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:¥Q>acegrep "say" sayno.txt | acegrep 'no'
but you will have to say no this time.

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

C:¥Q>
```

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

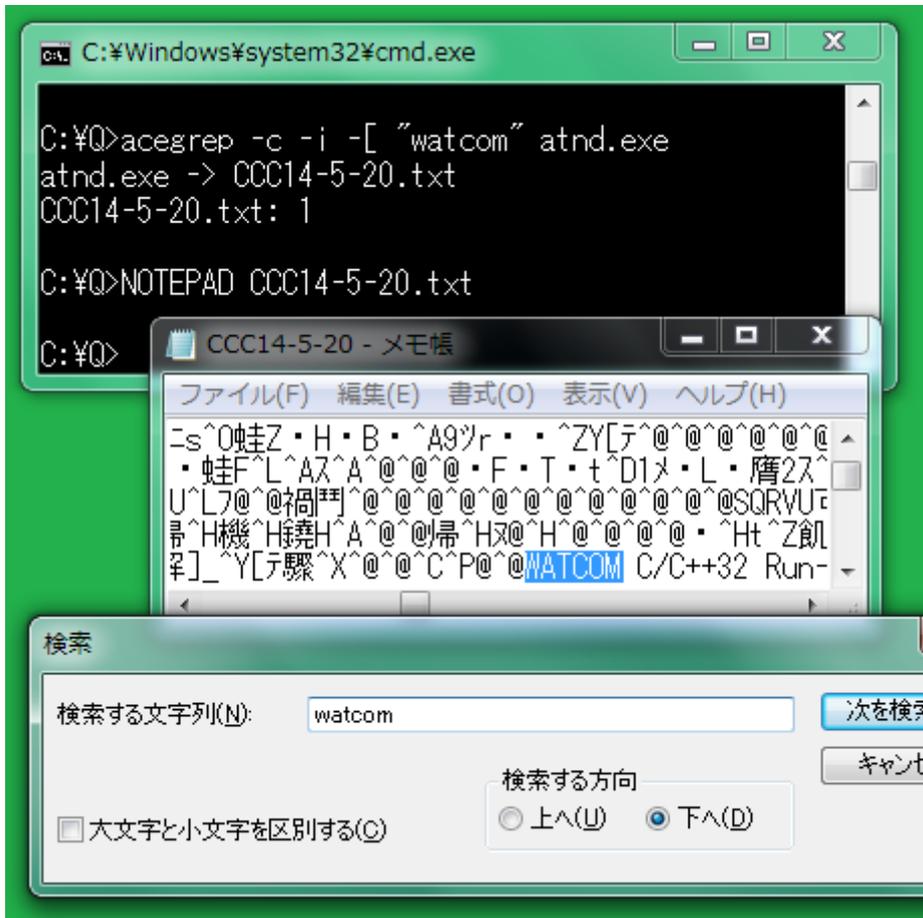
AceGrep は、いわゆる制御文字(数値)ではなく、 \backslash L, \backslash Z, \backslash [などの符号を出力できます。ただし、制御文字として、復帰、改行、タブは、それぞれ 13, 10, 9 の数値が出力されません。

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

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、 \backslash [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。AceGrep 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、AceGrep 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。

acegrep -c -i -["watcom" atnd.exe



バイナリファイルなら中止

acegrep コマンド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、AceGrep 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

AceGrep は、最後に入力された文字列を DIR コマンドにわたします。AceGrep 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、AceGrep 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。AceGrep 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

AceGrep 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。AceGrep 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。

```
C:\Windows\system32\cmd.exe
C:\>acegrep -c -u "[Ff]ree" "Psalms.txt poems.txt sayno.txt"
Psalms.txt: 5
poems.txt: 4
sayno.txt: 2

C:\>acegrep -c -u -r "[Ff]ree" "[Pps]*.txt"
C:\>Psalms.txt: 5
C:\>patterns.txt: 0
C:\>poems.txt: 4
C:\>sayno.txt: 2
C:\>text\patterns.txt: 0
C:\>text\poems.txt: 4
C:\>text\sayno.txt: 2

C:\>acegrep -c -r "[Ff]ree" "Psalms.txt poems.txt sayno.txt"
C:\>Psalms.txt: 5
C:\>poems.txt: 4
C:\>sayno.txt: 2
C:\>text\poems.txt: 4
C:\>text\sayno.txt: 2
```

AceList

DIR コマンドと異なり、ls コマンドは、ファイル名の大文字と小文字を区別します。ls コマンドと同様に、AceGrep に内蔵の AceList は、ファイル名の大文字と小文字を区別できます。acegrep コマンドのパラメータとして、--in が付加された場合、AceGrep は、DIR コマンドも ls コマンドも実行しないで、AceList で fileName.txt を作成します。

サブディレクトリも検索

AceGrep 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

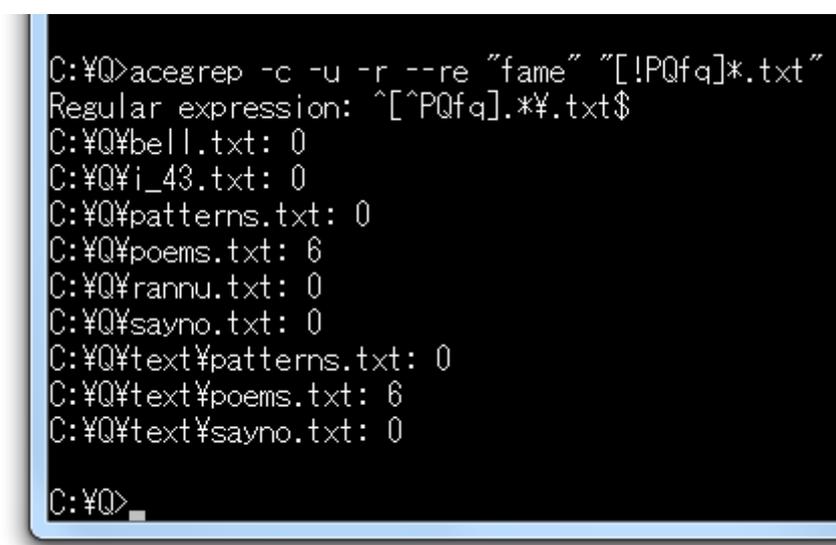
DIR コマンドは、コマンドプロンプトの内部コマンドですが、ls コマンドは、シェル(bash,

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、AceGrep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、AceGrep 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。

```
acegrep -c -u -r --re "fame" "[!f]*.txt"
```



```
C:¥Q>acegrep -c -u -r --re "fame" "[!PQfq]*.txt"
Regular expression: ^[!PQfq].*¥.txt$
C:¥Q¥bell.txt: 0
C:¥Q¥i_43.txt: 0
C:¥Q¥patterns.txt: 0
C:¥Q¥poems.txt: 6
C:¥Q¥rannu.txt: 0
C:¥Q¥sayno.txt: 0
C:¥Q¥text¥patterns.txt: 0
C:¥Q¥text¥poems.txt: 6
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、AceGrep 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。AceGrep 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、行頭を意味します。\$は、行末を意味します。

```
acegrep -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。

```
acegrep -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、AceGrep 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。

```
acegrep -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、AceGrep 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。

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

```
C:¥Q>acegrep -c -u -r "[Ee]ternal" poems.txt
C:¥Q¥poems.txt: 2
C:¥Q¥text¥poems.txt: 2

C:¥Q>acegrep -c -u -g "[Ee]ternal" poems.txt
C:¥Q¥poems.txt: 2

C:¥Q>acegrep -c -u -g -p- "[Ee]ternal" poems.txt
poems.txt: 2

C:¥Q>acegrep -c "[Ee]ternal" poems.txt
poems.txt: 2

C:¥Q>
```

-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、AceGrep 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、AceGrep は、まず、**バッチファイルと同等な処理**を実行します。次に、AceGrep 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。バッチファイルとして、LS_R_1.bat を作成するには、パラメータとして、--bat を付加してください。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:¥AceGrep¥LS_R_1 を入力して、Enter を押してください。【表示】標準出力は、1>であり、標準入力は、0<です。

C:¥AceGrep¥LS_R_1

```

C:\> C:\Windows\system32\cmd.exe
C:¥Q> acegrep -c -u -r --bat "%sgem" poems.txt
C:¥Q¥poems.txt: 4
C:¥Q¥text¥poems.txt: 4
C:¥Q> C:¥AceGrep¥LS_R_1
C:¥Q> ls -R -1 1>fileName.txt
C:¥Q> TYPE fileName.txt
.:
FNL1-16-21.txt
FNL2-51-25.txt
FNL3-51-36.txt
Psalms.txt
0.txt

```

AceList に多数のファイルをわたす

-u および -g が指定された場合も、-u および -r が指定された場合も、AceGrep は、引用符で囲まれた多数のファイル进行处理できませんが、AceGrep に内蔵の AceList は、空白で分離されており、引用符で囲まれている多数のファイルを順次、処理できます。

AceList はシェルのワイルドカードを処理できる

AceList は、Windows の C# で書かれていますが、コマンドプロンプトのワイルドカードを利用しません。パラメータとして、--in を付加した場合、acegrep コマンドにシェルのワイルドカードを付加してください。シェルのワイルドカードが正しい正規表現に変換されたか確認するには、--re を付加してください。

シェルのワイルドカード	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

```
C:\Windows\system32\cmd.exe
C:\>acegrep -c -j -u -r --bat "%sgem" poems.txt
C:\>%Q%poems.txt: 4
C:\>%Q%text%poems.txt: 4

C:\>C:\AceGrep%LS_R_1

C:\>ls -R -1 1>FNL3-51-36.txt

C:\>TYPE FNL3-51-36.txt
.:
FNL1-16-21.txt
FNL2-51-25.txt
FNL3-51-36.txt
Psalms.txt
Q.txt
```

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、acegrep コマンドのパラメータとして、`--ce` を付加してください。

```
acegrep --ce -r "fileNameListText" "*.cs"
```

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`(マイナスイエル)を付加してください。`-u` が指定されていない場合、AceGrep 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。

```
acegrep -l -r "fileNameListText" "*.cs"
```

```

C:\AceGrep>acegrep --ce -r "fileNameListText" "*.cs"
C:\AceGrep\AceGrep\AceGrep\AceList.cs: 3
C:\AceGrep\AceGrep\AceGrep\Grep.cs: 23
C:\AceGrep\AceGrep\AceGrep\UnixNotationAdapter.cs: 8

C:\AceGrep>acegrep -l -r "fileNameListText" "*.cs"
C:\AceGrep\AceGrep\AceGrep\AceList.cs
C:\AceGrep\AceGrep\AceGrep\Grep.cs
C:\AceGrep\AceGrep\AceGrep\UnixNotationAdapter.cs

C:\AceGrep>

```

カウントされなかったファイル

-Lが指定された場合、AceGrep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、AceGrepは、[Windowsに1を返している](#)場合があります。

acegrep -L -r ".*P" "*.cs"

```

C:\AceGrep>acegrep -L -r ".*P" "*.cs"
C:\AceGrep\AceGrep\AceGrep\AceList.cs
C:\AceGrep\AceGrep\AceGrep\Elapse.cs
C:\AceGrep\AceGrep\AceGrep\Error.cs
C:\AceGrep\AceGrep\AceGrep\Numbered.cs
C:\AceGrep\AceGrep\AceGrep\Program.cs
C:\AceGrep\AceGrep\AceGrep\Singleton.cs
C:\AceGrep\AceGrep\AceGrep\TimeName.cs
C:\AceGrep\AceGrep\AceGrep\UnixNotationAdapter.cs
C:\AceGrep\AceGrep\AceGrep\WildcardToRegularExpression.cs
C:\AceGrep\SideAce\SideAce\Error.cs
C:\AceGrep\SideAce\SideAce\Form1.Designer.cs
C:\AceGrep\SideAce\SideAce\Numbered.cs
C:\AceGrep\SideAce\SideAce\Version.cs
C:\AceGrep\SideAce\SideAce\Properties\Settings.Designer.cs

C:\AceGrep>

```

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、AceGrep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。パス名を表示しない命令として、`-p-` は、分解されません。`--bat`, `--ce`, `--in`, `--label=`, `--re` 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。

```
C:\>acegrep -c -u -r -j --re "%sgem" poems.txt
Regular expression: ^poems%.txt$
C:\>%Q%poems.txt: 4
C:\>%Q%text%poems.txt: 4

C:\>acegrep -curj --re "%sgem" poems.txt
Regular expression: ^poems%.txt$
C:\>%Q%poems.txt: 4
C:\>%Q%text%poems.txt: 4

C:\>_
```

ファイルが発見されない

`-b-` が指定されていない場合、AceGrep 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、AceGrep 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、AceGrep 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。

fileLength < TOO_SMALL

待機時間を調節できる

AceGrep 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。

```
C:¥Q>TYPE gone.bat
@ECHO OFF
funbyo
acegrep -c -b%1 "stone" poem.txt
funbyo

C:¥Q>gone 80
51:03ファイルが見つかりません
fileLength < TOO_SMALL
51:12
C:¥Q>_
```

経過時間

AceGrep 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、AceGrep のミリ秒の数値は、測定が不正確です。

acegrep -c -@ "Loyalty" poems.txt

計数

パラメータとして、--tick が付加された場合、AceGrep は、C#の Stopwatch クラスを利用して、計数(tick)を表示します。ミリ秒および計数を併用できます。

```

C:¥Q>acegrep -c -@ "Loyalty" poems.txt
8 milliseconds elapsed before fileName.txt was written.
poems.txt: 1
8 milliseconds elapsed from start to finish.

C:¥Q>acegrep -c --tick "Loyalty" poems.txt
260951 ticks
poems.txt: 1
293746 ticks

C:¥Q>

```

-bで時間が短縮される

-bが指定された場合、AceGrepは、既存の fileName.txt を読み込みます。この場合、AceGrep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。**

```

C:¥Q>acegrep -c -r -@ --tick "stone" "p*.txt"
260421 ticks
8 milliseconds elapsed before fileName.txt was written.
C:¥Q¥patterns.txt: 0
C:¥Q¥poems.txt: 15
C:¥Q¥Psalms.txt: 8
C:¥Q¥text¥patterns.txt: 0
C:¥Q¥text¥poems.txt: 15
341609 ticks
10 milliseconds elapsed from start to finish.

C:¥Q>acegrep -c -b- -@ --tick "stone"
1896 ticks
One millisecond elapsed.
C:¥Q¥patterns.txt: 0
C:¥Q¥poems.txt: 15
C:¥Q¥Psalms.txt: 8
C:¥Q¥text¥patterns.txt: 0
C:¥Q¥text¥poems.txt: 15
36174 ticks
2 milliseconds elapsed from start to finish.

C:¥Q>

```

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、AceGrep 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。

acegrep -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、AceGrep 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、AceGrep 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、AceGrep 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、AceGrep は、1 を返します。エラーの場合、AceGrep は、2 を返します。

ECHO コマンド

AceGrep 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%

```
C:¥Q>acegrep -L "Loyalty" poems.txt
C:¥Q>ECHO %ERRORLEVEL%
0
C:¥Q>acegrep -L "loyalty" poems.txt
poems.txt
C:¥Q>ECHO %ERRORLEVEL%
1
C:¥Q>acegrep -L "[Lloyalty" poems.txt
compiling(re): /[Lloyalty/
C:¥Q>ECHO %ERRORLEVEL%
2
C:¥Q>acegrep -L "[L]oyalty" poems.txt
C:¥Q>ECHO %ERRORLEVEL%
0
C:¥Q>
```

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、AceGrep 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。

acegrep -L -q "[Lloyalty" poems.txt

```
C:¥Q>acegrep -L -q "[Lloyalty" poems.txt
C:¥Q>ECHO %ERRORLEVEL%
1
C:¥Q>acegrep -L -q "[L]oyalty" poems.txt
C:¥Q>ECHO %ERRORLEVEL%
0
C:¥Q>
```

何も出力しない

何行かマッチしたにもかかわらず、AceGrep 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:\Windows\system32\cmd.exe
C:\>acegrep -L "[Lloyalty]" poems.txt > NUL
compiling(re): /[Lloyalty/

C:\>ECHO %ERRORLEVEL%
2

C:\>acegrep -L -q "[Lloyalty]" poems.txt > NUL

C:\>ECHO %ERRORLEVEL%
1

C:\>acegrep -L -q "[LI]oyalty" poems.txt > NUL

C:\>ECHO %ERRORLEVEL%
0

C:\>_
```

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。

acegrep -c -x "Hope." poems.txt

カレットおよびドル

-x が指定された場合、AceGrep 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。

```
C:\>acegrep -c "Hope." poems.txt
poems.txt: 4

C:\>acegrep -c -x "Hope." poems.txt
poems.txt: 2

C:\>
```

マッチした行を出力

AceGrep 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。

```
acegrep -n "realm" poems.txt
```

```
C:¥Q>acegrep -n "realm" poems.txt
298: Its infinite realms contain
2491: The realm of you.
4870: That devastated childhood's realm,
4956: The Caspian has its realms of sand,
4957: Its other realm of sea;
6187: Oftener through realm of briar
C:¥Q>_
```

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、`-I`(マイナスイエル)を付加してください。

```
acegrep -H -r "Regex" "*.cs"
```

	カウントする	カウントしない
行を出力する		<code>-H</code>
行を出力しない	<code>--ce</code>	<code>-I</code>

周辺の間を出力

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間を出力できます。さもないければ、[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、AceGrep 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。

acegrep -A2 "barefoot" poems.txt

```
C:¥Q>acegrep -A2 "barefoot" poems.txt
There, _ sandals for the barefoot;
There, _ gathered from the gales,
Do the blue havens by the hand
--
Yet when a child, and barefoot,
I more than once, at morn,
Have passed, I thought, a whip-lash
--
C:¥Q>
```

-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、AceGrep 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。

acegrep -B3 "[Cc]lasp" poems.txt

```
C:¥Q>acegrep -B3 "[Cc]lasp" poems.txt
Where this is said to be
But just the primer to a life
Unopened, rare, upon the shelf,
Clasped yet to him and me.
--
It can't be dying, _ it's too rouge, _
The dead shall go in white.
So sunset shuts my question down
With clasps of chrysolite.
--
C:¥Q>
```

-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。

acegrep -C4 "^Line-384" LineNo.txt

```
C:¥Q>acegrep -C4 "^Line-384" LineNo.txt
Line-380: 4 lines to Line-384
Line-381: 3 lines to Line-384
Line-382: 2 lines to Line-384
Line-383: 1 line to Line-384
Line-384
Line-385: 1 line from Line-384
Line-386: 2 lines from Line-384
Line-387: 3 lines from Line-384
Line-388: 4 lines from Line-384
--
C:¥Q>
```

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、AceGrep 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

AceGrep 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。

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

```
C:¥Q>atnd -p1514 Psalms.txt
Psalms:94:22: But the LORD is my defence; and my God is the rock of my refuge.

C:¥Q>acegrep -H -n "rock of my refuge" Psalms.txt
Psalms.txt: 1514: Psalms:94:22: But the LORD is my defence; and my God is the rock of my refuge.

C:¥Q>ECHO /> separate.txt

C:¥Q>acegrep -H -n -S "rock of my refuge" Psalms.txt
Psalms.txt/1514/Psalms:94:22: But the LORD is my defence; and my God is the rock of my refuge.

C:¥Q>
```

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。

```
acegrep -n -T6 "[Rr]ock" poems.txt
```

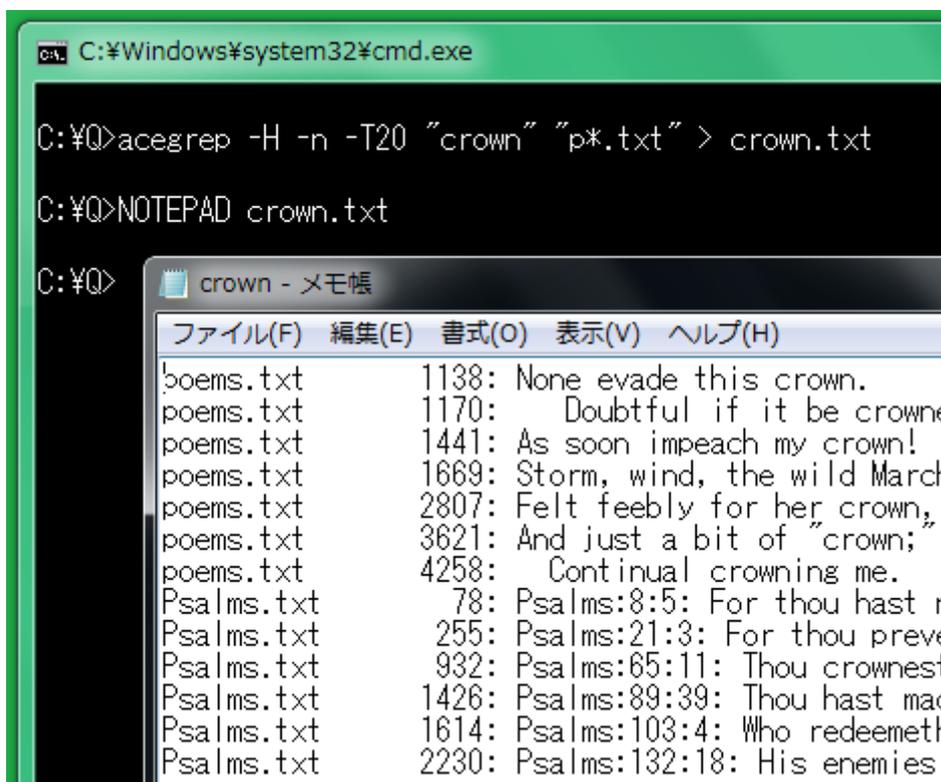
```
C:¥Q>acegrep -n -T6 "[Rr]ock" poems.txt
  118: The smitten rock that gushes,
 2615: Her little rocking-chair to draw,
 2907: That ever rocked a child.
 3355: The wind begun to rock the grass
 3933: Rocked softer, to and fro_
 4549: Late, my acre of a rock
 6041: Will the frock I wept in

C:¥Q>
```

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。

```
acegrep -H -n -T20 "crown" "p*.txt"
```



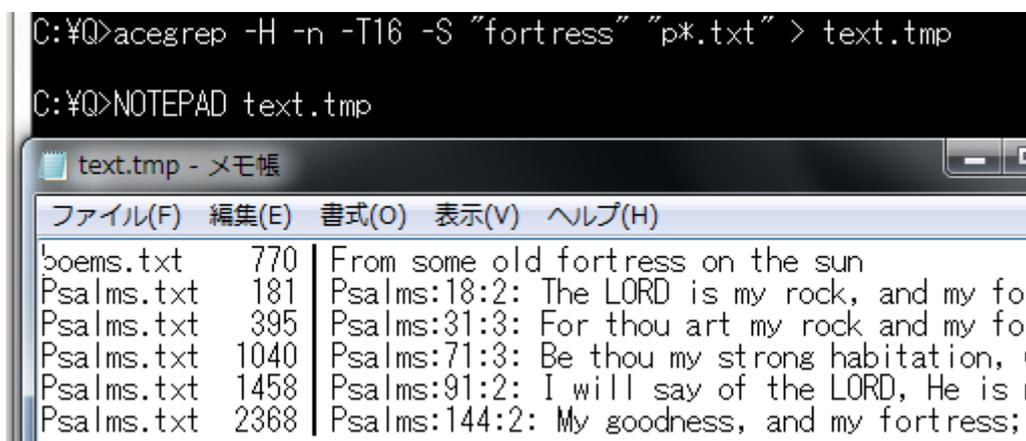
```
C:\Windows\system32\cmd.exe
C:\>acegrep -H -n -T20 "crown" "p*.txt" > crown.txt
C:\>NOTEPAD crown.txt
C:\>
```

ファイル(F)	編集(E)	書式(O)	表示(V)	ヘルプ(H)
boems.txt	1138:	None	evade	this crown.
poems.txt	1170:		Doubtful	if it be crowne
poems.txt	1441:		As soon	impeach my crown!
poems.txt	1669:		Storm, wind,	the wild March
poems.txt	2807:		Felt feebly	for her crown,
poems.txt	3621:		And just a bit	of "crown;"
poems.txt	4258:		Continual	crowning me.
Psalms.txt	78:	Psalms:8:5:	For thou	hast m
Psalms.txt	255:	Psalms:21:3:	For thou	preve
Psalms.txt	932:	Psalms:65:11:	Thou	crownest
Psalms.txt	1426:	Psalms:89:39:	Thou	hast mad
Psalms.txt	1614:	Psalms:103:4:	Who	redeemeth
Psalms.txt	2230:	Psalms:132:18:	His	enemies

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、AceGrep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、AceGrepに-Sで読み込ませた例

```
acegrep -H -n -T16 -S "fortress" "p*.txt"
```



```
C:\>acegrep -H -n -T16 -S "fortress" "p*.txt" > text.tmp
C:\>NOTEPAD text.tmp
C:\>
```

ファイル(F)	編集(E)	書式(O)	表示(V)	ヘルプ(H)
boems.txt	770		From some	old fortress on the sun
Psalms.txt	181	Psalms:18:2:	The LORD	is my rock, and my for
Psalms.txt	395	Psalms:31:3:	For thou	art my rock and my for
Psalms.txt	1040	Psalms:71:3:	Be thou	my strong habitation, v
Psalms.txt	1458	Psalms:91:2:	I will	say of the LORD, He is n
Psalms.txt	2368	Psalms:144:2:	My	goodness, and my fortress;

マッチしない行を出力

マッチしない行を出力するには、acegrep コマンド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。

acegrep -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、AceGrep 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	

```
C:¥Q>acegrep -v "Psalms" Psalms.txt
King James Version of 1611

C:¥Q>
```

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、AceGrep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

```
C:¥Q>acegrep "bird" poems.txt | acegrep 'sing'
At half-past three a single bird
Before the blackbirds sing.

C:¥Q>acegrep "bird" poems.txt | acegrep -k "sing"
At half-past three a single bird
Before the blackbirds sing.

C:¥Q>_
```

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 を押しながらかを押してください。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。

```
acegrep -m3 "bird" "p*.txt"
```

```
C:\>acegrep -m3 "bird" "p*.txt"
And blushing birds go down to drink,
And birds of foreign tongue!
The birds jocosely sung;
Psalms:11:1: In the LORD put I my trust: How say ye to my soul, Flee as a bird to
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:\>
```

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。C#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、AceGrep 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。acegrep コマンドのパラメータとして、-#を付加することで、CHCP 実行結果の一部を出力できます。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。

acegrep -# "a" bell.txt



```
C:\¥Q>acegrep -# "a" bell.txt
932
C:\¥Q>
```

各行に文字コード

grep と異なり、AceGrep 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、AceGrep 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、AceGrep 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

AceGrep 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、AceGrep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。

acegrep -8 “立琴” Samuel.txt

```
C:\%Q>acegrep -8 “立琴” Samuel.txt
10:1その時サムエルは油のびんを取って、サウルの頭に注ぎ、彼に口づけして言った、「主はあなたに油を注いで、その民イスラエルの君とされたではありませんか。あなたは主の民を治め、周囲の敵の手から彼らを救わなければならない。主があなたに油を注いで、その嗣業の君とされたことの、しるしは次のとおりです。 10:2あなたがぎょう、わたしを離れて、去って行くとき、ベニヤミンの領地のゼルザにあるラケルの墓のかたわらで、ふたりの人に会うでしょう。そして彼らはあなたに言います、『あなたが捜しに行かれたるばは見つかりました。いま父上は、ろばよりもあなたがたの事を心配して、「わが子のことは、どうしよう」と言っておられます』。 10:3あなたが、そこからなお進んで、タボルのかしの木の所へ行くと、そこでベテルに上って神を拝もうとする三人の者に会うでしょう。ひとりには三頭の子やぎを連れ、ひとりには三つのパンを携え、ひとりには、ぶどう酒のはいった皮袋一つを携えている。 10:4彼らはあなたにあいさつし、二つのパンをくれるでしょう。あなたはそれを、その手から受けなければならない。 10:5その後、あなたは神のギベアへ行く。そこはペリシテびとの守備兵のいる所である。あなたはその所へ行って、町にはいる時、立琴、手鼓、笛、琴を執る人々を先に行かせて、預言しながら高き所から降りてくる一群の預言者に会うでしょう。 10:6その時、主の雲があなたの上にも
```

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。パラメータとして、`--encoding=CP437` を付加できます。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。

acegrep --encoding=CP1252 “gem” poems.txt

ISO-8859-1

英語のファイルである場合、パラメータとして、`--encoding=ISO-8859-1` を付加できます。=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、AceGrep は、ラテン文字-1 のファイルを入力します。

acegrep --encoding=ISO-8859-1 “gem” poems.txt

認識される文字コード	パラメータ	説明
ASCII		
EUC-JP	-Ke	UNIX の日本語
EUC_JP	-Ke	UNIX の日本語
ISO-8859-1		Latin-1, ラテン文字-1
ISO8859_1		Latin-1,ラテン文字-1
JIS		ISO-2022-JP, 日本語の電子メール
MS932	-Kw	Windows-31J, 日本語版の Windows
Shift_JIS	-Ks	日本語
SJIS	-Ks	日本語
Unicode		UTF-16
US_ASCII		
UTF-16		
UTF-32		
UTF-7		
UTF-8	-Ku	
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。

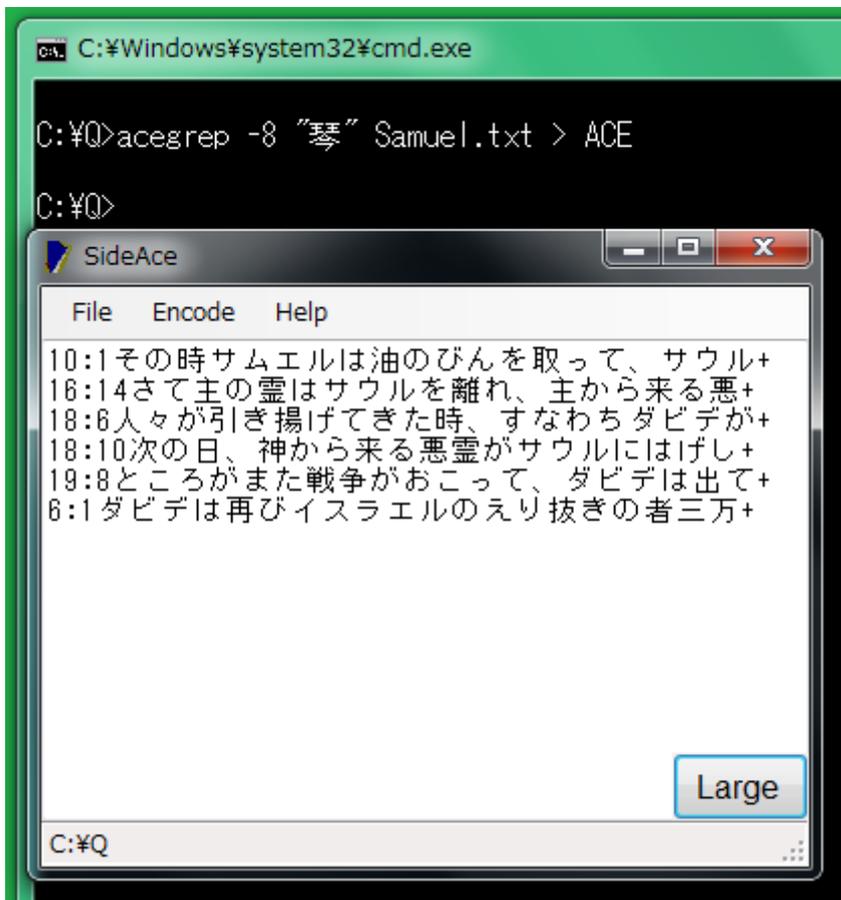
SideAce ウィンドウ

AceGrep に C#版の SideAce を添付しました。これは、Windows フォームアプリケーションであり、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

AceGrep 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。

acegrep -8 “琴” Samuel.txt > ACE



連携

SideAce と連携するには、acegrep コマンド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。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。

acegrep --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、SideAce のカレントディレクトリが表示されています。カレントディレクトリのことを作業フォルダーとも言います。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。

大きいウィンドウ

SideAce ウィンドウを大きくするには、Large ボタンを押してください。さもなければ、Help メニューから Large を選択してください。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。サイズが変更されたとき、SideAce は、ウィンドウの幅から表示桁数を算出します。

ダブルクリック

SideAce のショートカットを作成してコピーする手順を示します。

1. 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。
2. C:\%AceGrep のフォルダをクリックして、内容を表示してください。

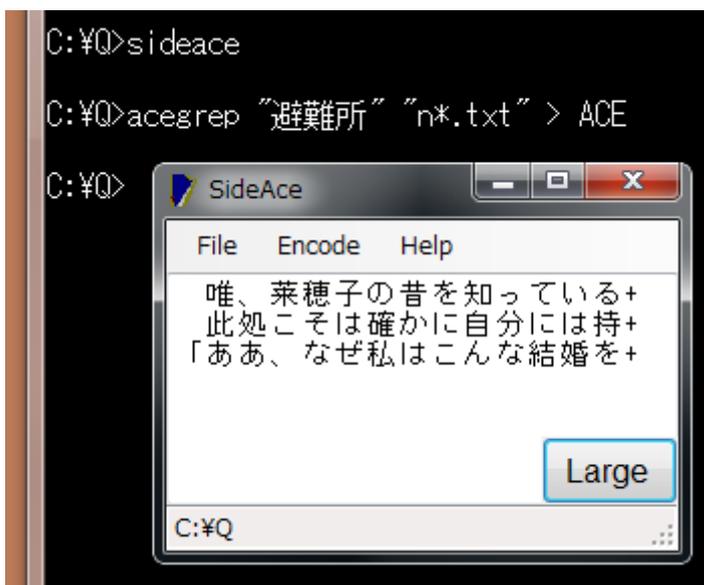


3. SideAce のアイコンを右クリックしてください。 SideAce
4. メニューから[ショートカットの作成]を選択してください。
5. SideAce のショートカットをコマンドプロンプトの作業フォルダー(C:\%Q)にドラッグしてドロップしてください。
6. SideAce のショートカットを右クリックしてください。

7. メニューからプロパティを選択してください。
8. 作業フォルダーに正しいフォルダ名 (C:¥Q) を入力してください。
9. OK ボタンを押してください。
10. SideAce のショートカットをダブルクリックしてください。
11. SideAce ウィンドウの下部にあるステータスバーに作業フォルダーが表示されていることを確認してください。

sideace コマンド

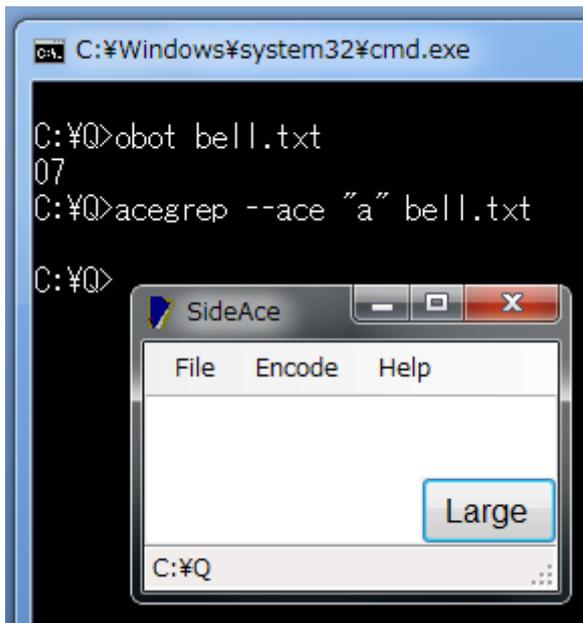
コマンドプロンプトに sideace コマンドを入力することで、SideAce を起動できます。



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。

```
acegrep --ace "a" bell.txt
```



sideace コマンドのパラメータ

SideAce.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。

sideace -VC

```
C:\>sideace -VC
C:\>TYPE version.txt
SideAce C# Version 1.01

SideAce -c4 # Move the SideAce window to Corner-4
SideAce -e2 # 1: EUC-JP. 2: Shift_JIS. 3: UTF-8
SideAce -p # PDF help
SideAce -V # version
SideAce -VC # version with comments
C:\>
```

スクロール

[画面バッファのサイズ](#)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。

acegrep -8 "ダビデ" Samuel.txt | MORE

クリップボード

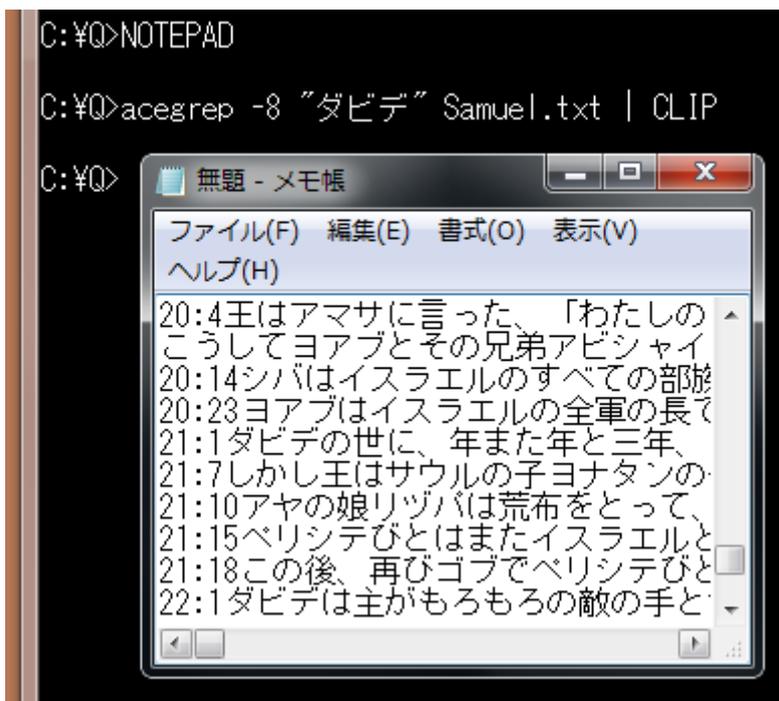
どの Windows も最初からクリップボードを用意しています。コマンドプロンプトのウィンドウとは独立したメモ帳ウィンドウに、クリップボードからコマンドの出力を貼り付けることができます。

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。

acegrep -8 "ダビデ" Samuel.txt | CLIP



GNU Win32

GNU coreutils Windows をインターネットで検索したら、Download のウェブページが発見されます。下記の無料アプリケーションを入手できます。

ダウンロード

Setup をダウンロードしました。これは、インストーラーです。

coreutils-5.3.0

インストール

32 ビット版 Windows 7 のパソコンに coreutils-5.3.0 をコピーして、ダブルクリックしてインストールしました。ls コマンドもインストールされました。wc コマンドもインストールされました。

パスを通す

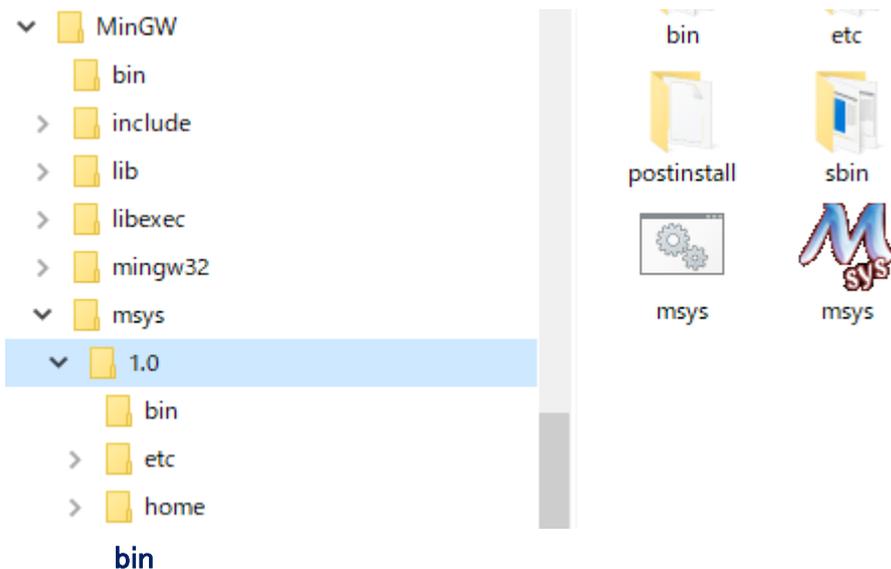
MinGW をインストールした場合、ls コマンド、wc コマンドもインストールされます。コマンドプロンプトでこれらを利用するには、パスを通す必要があります。

msys のアイコン

ls コマンドは、どこにあるのでしょうか？ エクスプローラーの右上にある検索ボックスに、ls.exe を入力して、コンピュータの全体にわたって検索できますが、非常に時間がかかります。MinGW のフォルダで、まず msys のアイコンをさがしてください。手順の例を示します。

1. msys のアイコンをさがしてください。
2. bin のフォルダの内容を表示してください。ls というアプリケーションがあります。
3. ls を右クリックしてください。
4. メニューから、プロパティを選択してください。
5. 場所の右をドラッグして選択してください。
6. Ctrl を押しながら、C を押してください。クリップボードにコピーされます。
7. メモ帳などのテキストエディタで新しいファイルに貼り付けることができます。
8. [システム、詳細、環境変数](#) のキーワードを手掛かりに、Path を編集する入力ボックスを開いてください。
9. Windows 10 の場合、[テキストの編集] ボタンを押してください。
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icolonは、ディレクトリどうしを区切る記号です。
11. セミicolonおよび ls コマンドの場所(C:¥MinGW¥msys¥1.0¥bin)を入力してください。

12. 各ウィンドウで OK を押してください。



上図には、bin が 2 個あります。msys のアイコンの下位ディレクトリとして、bin の内容を表示してください。ls があります。

