1　ビルトインコマンド（自作コマンド）の仕様

1.1　ディレクトリの管理機能

cd コマンド

ディレクトリ名が指定された場合は、そのディレクトリをカレントディレクトリに設定する。ディレクトリ名が指定されなかった場合は、環境変数 HOME に指定されたディレクトリをカレントディレクトリに設定する。ディレクトリの指定は絶対パス、相対パスのどちらでも可能である。また相対パスを用いてディレクトリ名を指定する際、相対パスの起点を、「∼」(環境変数 HOME に指定されたディレクトリ)、「 . 」(カレントディレクトリ)、「 .. 」(1 つ上の階層のディレクトリ) といった記号で指定することができる。また 「 .. 」は「 ../../../ 」のように使うことで 2 つ以上上の階層のディレクトリを指定できる。ただし、階層を上にたどっていく過程でルートディレクトリに到達した場合は、相対パスの起点はルートディレクトリとなる。

cd　[ディレクトリ名]

pushd コマンド

ディレクトリスタックへカレントディレクトリを保存する。

pushd

dirs コマンド

現在のディレクトリスタックを、スタックの上 (最初に取り出される方) から順に表示する。

dirs

popd コマンド

現在のディレクトリスタックの 1 番上のディレクトリをカレントディレクトリに設定する。また取り出したディレクトリはスタックから削除し、その他のスタックの要素をそれぞれ上に移動させる。スタックの要素がない場合はその旨を表示する。

popd

1.2 ヒストリー機能

history コマンド

シェルを起動してから実行したコマンドをヒストリとして実行した順番とともに保存しておき、それを古いもの (番号の小さいもの) から順に表示する。なお、ヒストリとして保存するコマンドの数は３２個とし、コマンドとともに保持する順番はシェル起動時を基準とする。また、３２個を超えた場合は、1 番新しい (番号の大きい) ものから３２個目までを保存する。またコマンドに何も入力されなかった場合 (厳密には改行が入力されているが) は保存しない。さらに、1.2 節、1.2 節でそれぞれ説明する !! コマンドと !string コマンドは、実際に実行したコマンドを保存する。(例えば、 1 つ前に実行したコマンドが ls コマンドだった場合、!! コマンドを実行すると、ヒストリには!! ではなく ls が保存される。)

```
history
```

!!コマンド

1.2 節の history コマンド同様、シェルを起動してから実行したコマンドをヒストリとして実行した順番とともに保存しておき、 1 つ前に実行したコマンドを再度実行する。また再度実行したコマンドもヒストリに追加する。ただし実行できなかった場合 (1 つ前のコマンドが存在しなかった場合) は追加しない。

```
!!
```

!string コマンド

1.2 節の history コマンド同様、シェルを起動してから実行したコマンドをヒストリとして実行した順番とともに保存しておき、string(1 文字以上の任意の文字列) で始まる最新のコマンドを再度実行する。ヒストリとして保存されてる履歴の範囲内に、string で始まるコマンドがない場合にはエラーメッセージを出力する。また再度実行したコマンドもヒストリに追加する。ただし実行できなかった場合 (ヒストリとして保存されている範囲内に一致するものがなかった場合) は追加しない。

```
!string
```

!n コマンド

簡易版シェルを起動してから、n 番目に実行したコマンドを再度実行する。ヒストリリストに n 番目が保存されていない場合は実行できない。n は数字とする。

```
!n
```

!-n コマンド

n 個前に実行したコマンドを再度実行する。ヒストリリストに n 個前のコマンドが保存されていない場合は実行できない。n は数字。

```
!-n
```

1.3 ワイルドカード機能

*****

コマンドにある「 * 」をカレントディレクトリ内にあるすべてのファイル名に置き換える。ディレクトリは置き換えないようにした。「*」は単独で使われる、前後に空白類があるものとする (「 ./* 」のような使われ方は想定していない)。また「 * 」が 2 つ以上使われることは想定していないため、 2 つ目からは置き換えられない。

1.4 プロンプト機能

prompt コマンド

strings によって指定された文字列にプロンプトを変更する。文字列が指定されなかった場合は、デフォルトの文字列「 Command : 」に変更する。また指定された文字列に、タブや 2 つ以上の空白が含まれた場合は 1 つの空白とみなす。

```
prompt [strings]
```

1.5 スクリプト機能

仕様

作成したシェル mysh が、script_file に記述されたコマンドを実行する。ファイルには 1 行に 1 つのコマンドが記述されているとする。ファイルに exit1 が記述されていなくても正常終了する。また通常のシェルは、標準入力が端末でない場合プロンプトを出力しないが、本課題では出力する。ファイルはリダイレクトによって読み込まれるとする。また、コマンドごと改行は行われない。

1.6 エイリアス機能

alias コマンド

コマンド 2 の別名を command1 として設定する。以後、command1 が入力された場合、command2 に置き換えて実行する。また引数なしで alias コマンドを実行した場合は、現在登録されている alias の一覧を表示する。ただし、command1 には空白類を含むことができない。command2 には空白類を含むことができる。また command1 が登録済みの場合は、新しく実行したほうに更新する。

alias [command1 command2]

unalias コマンド

alias コマンドによって別名設定された command1 を解除する。

unalias [command1]

1.7 その他機能

grep コマンド

指定したファイル (files) の中で、指定した文字列 (string) が含まれる行の文と行番号とファイル名を表示する。ファイルは複数指定可能で、string は空白類を含まない文字列である。

grep [string] [files]

cat コマンド

指定したファイルの中身を表示する。ファイルは複数指定可能である。引数が指定されない場合は、使用方法を表示する。

cat [files]