Lecture 10

More Perl

COP 3344 Introduction to UNIX

Fall 2007

Acknowledgment: These slides are modified versions of Prof. Sudhir Aggarwal's slides

system

- Shell executes the command that is given as the argument
  - `system("command")`

pprog1
```
#!/usr/bin/perl -w

 system("rm *.bak ");
```

```
$ ls
file1.bak  file2.txt.bak  pprog1
$ ./pprog1
$ ls
pprog1
```

File I/O

- In the code here, `FILE1` and `FILE2` are "handles" to files
  - A file needs to be opened before it is used
    - The '<' indicates opening for reading
    - The '>>' indicates opening for appending
    - A '>' indicates opening for writing
  - `<FILE1>` reads in a line from the file pointed to by the handle `FILE1`
  - *print* takes as its first argument a file handle
    - This handle defaults to `STDOUT`

pprog2
```
#!/usr/bin/perl -w

open(FILE1, '<', 'filerd');
$line=<FILE1>;
print $line;
close(FILE1);

open(FILE2, '>>', "fileap");
print FILE2 $line;
close(FILE2);
```

filerd
```
Test file
```

```
$./pprog2
Test file
$./pprog2
Test file
$cat fileap
Test file
Test file
```

while Loop

- A while loop has the following form
  ```
  while(condition){
  statements
  }
  ```
  - The statements are repeatedly executed as long as the condition remains true
  - While reading using a file handle, the condition remains true until the end of file (EOF) is reached

pprog4
```
#!/usr/bin/perl -w

open(FILE1, '<', 'pprog1');
while(<FILE1>){
    print $_;
}
close(FILE1);
```

pprog1
```
#!/usr/bin/perl -w

 system("rm *.bak ");
```

```
$./pprog4
#!/usr/bin/perl -w

 system("rm *.bak ");
```

Arrays

- Use @ to indicate an array
  - *`#ArrayName`* refers to the number of elements in the array
  - `split(`*`separator – pattern, string`*`)` splits a string into fields
    - The default separator is white space

pprog5
```
#!/usr/bin/perl -w

@OPTION=("-a", "-r", "-w");
print $OPTION[0] . "  " .
$OPTION[$#OPTION] . "\n";

open(FILE1, '<', 'filerd');
while(<FILE1>){
  @fields = split;
  $count=0;
  while($count <= $#fields){
    print $fields[$count] . " ";
    $count = $count + 1;
  }
  print "\n";
}
close(FILE1);
```

filerd
```
Test file
```

```
$./pprog5
-a  -w
Test file
```

Command Line Arguments

- The array ARGV stores the command line arguments
  - #ARGV stores the last array index

pprog3
```
#!/usr/bin/perl -w

 if (!($#ARGV>=0)){
  print STDERR "Incorrect number of command line arguments\n";
  exit 1;
}
print $ARGV[0]."\n";
```

```
$ ./pprog3 qwer sdgf
qwer
$ ./pprog3
Incorrect number of command line arguments
```