PRINT Name: ______________________________ **LAB Section:** □□□

Test Version: `807` **One-Answer Multiple Choice** **218 Questions − 15 of 15%**

☞ Read **all** the words of these instructions and **both** sides (back and front) of all pages.
☞ Manage your time. Answer questions you know, first. One Answer per question.
☞ Put your Name and Lab on this Question Sheet. You may write or draw on this sheet.
☞ Use your full, unabbreviated name on the mark-sense form. Do not abbreviate your name.
☞ Put the three-digit **Test Version** above into both **NO. OF QUESTIONS** and **NO. OF STUDENTS**
☞ Fill in the bubbles with pencil only, no pen. Enter your NAME, Test Version, and answers.
☞ The answer to the questions below about reading/doing all these test instructions is: **`Jes`**

---

1. **Did you read all the words of the test instructions on page one?**
   a. **`Taip`** *(Yes - Lithuanian)* *b.* **`Sim`** *(Yes - Portuguese)*
   c. **`Igen`** *(Yes - Hungarian)* *d.* **`Jes`** *(Yes - Esperanto)*
   e. **`Tak`** *(Yes - Polish)*

2. **My three-digit Lab Section number is:**
   a. The timetable section number of my weekly 2-hour lab period.
   b. My lab room number, e.g. **`B182`**, **`B119`**, **`J218`**, **`CA418`**
   c. My lecture section number, e.g. **`010`** or **`020`**.
   d. The Test Version number printed in the top left corner.
   e. My lecture room number, e.g. **`T117`**

3. What is the output on your screen after these command lines:

```
echo one >x ; ln x y ; echo two >>y
sort x >y ; cat y
```

   a. **`two`** *b.* **`one`** followed by **`two`**
   c. no output *d.* **`two`** followed by **`one`**
   e. **`one`**

4. What is in the local variable **`$$`** ?
   a. **`$$`** is not a valid variable name
   b. the cpu cost of the current session, in dollars
   c. the command name of the previous command line
   d. the first argument of the previous command line
   e. the process ID of the current shell

5. How many arguments are passed to the command by the shell:

```
    <wc wc " 1 '2 3' 4 " 5 6 ' 7 " 8 " ' >wc 9
```

   a. **`5`** *b.* **`6`** *c.* **`2`** *d.* **`3`** *e.* **`4`**

6. If the file **`bat`** contained the word **`foo`**, what is the output on your screen after this: **`PATH=/bin/cat:/bin/who:/bin/ls ; cat bat`**
   a. no output on screen
   b. **`foo`**
   c. **`bat`**
   d. **`bash: cat: command not found`**
   e. **`cat: bat: No such file or directory`**

7. How many arguments are passed to the command by the shell:

```
    echo "cow "y " bat 'man x' " pig'a "hop' a b
```

   a. **`4`** *b.* **`5`** *c.* **`7`** *d.* **`6`** *e.* **`11`**

8. Which one of these names is usually a shell environment variable?
   a. **`Foobar`** *b.* **`foobar`** *c.* **`fooBar`**
   d. **`FooBar`** *e.* **`FOOBAR`**

9. What is the link count of file **`foo`** after these successful commands?

```
rm foo ; touch foo ; ln foo bar
cp bar x ; ln x y ; ln bar z
```

   a. **`3`** *b.* **`4`** *c.* **`1`** *d.* **`5`** *e.* **`2`**

10. If **`foo`** were a readable empty file, what is the output on your screen after this:

```
    PATH=/etc/passwd:/bin/ls:/bin/cat ; /bin/cat foo
```

   a. **`bash: ls: command not found`**
   b. no output on screen
   c. **`bash: cat: command not found`**
   d. **`bash: /bin/cat: command not found`**
   e. **`/bin/cat: foo: No such file or directory`**

11. If **`/bin/foo`** is a program that outputs **`dad`** and **`/usr/bin/foo`** is a program that outputs **`mom`** what is the output on your screen after this:

```
    PATH=/usr:/etc:/bin:/usr/bin ; foo
```

   a. **`mom`** followed by **`dad`**
   b. **`bash: foo: command not found`**
   c. **`dad`** followed by **`mom`**
   d. **`dad`**
   e. **`mom`**

12. How many arguments are passed to the command by the shell:

```
    echo 'It's a bird! No! It's a plane!'
```

   a. **`4`** *b.* **`5`** *c.* **`3`** *d.* **`1`** *e.* **`2`**

13. If `/bin/foo` is a program that outputs `mom` and `/usr/bin/foo` is a program that outputs `dad` what is the output on your screen after this:

```
    PATH=/bin/foo:/usr/bin/foo:/usr ; foo
```

a. `dad` followed by `mom`
b. `dad`
c. `bash: foo: command not found`
d. `mom` followed by `dad`
e. `mom`

14. If file `foo` occupies one disk block, how many disk blocks are in use after this:

```
    cp foo bar ; ln bar one ; cp one two ; cp one xxx
```

a. `3` *b.* `4` *c.* `2` *d.* `5` *e.* `1`

15. In an empty directory, what is the output on your screen after this:

```
    touch 1 2 3 ; cow="*" ; echo $cow
```

a. `*` *b.* `1 2 3` *c.* `$cow`
d. `"*"` *e.* `"1 2 3"`

16. Given this `ls -il` long listing:

```
    123 drwxr-xr-x 456 me me 789 Jan 1 1:00 dir
```

How many subdirectories lie immediately under `dir`?

a. `454` *b.* `787` *c.* `456` *d.* `789` *e.* `123`

17. How many files are touched? `touch '1  "2 3  '4'" '  5`

a. `2` *b.* `4` *c.* `1` *d.* `5` *e.* `3`

18. What is the link count of directory `z` after these successful commands?

```
    mkdir z ; mkdir z/a z/a/b z/a/c z/a/d
```

a. `5` *b.* `2` *c.* `3` *d.* `1` *e.* `4`

19. If directory `/a` contains these seven two-character names: `aa`, `ab`, `ac`, `ad`, `a?`, `a*`, `a.`, then which command removes *only* the single two-character name `a?` from the directory?

a. `rm /a/a?` *b.* `rm /a/a[*]` *c.* `rm /a/a*`
d. `rm /a/a\?` *e.* `rm /a?`

20. If I have a directory named `/1/2`, which action would increase its *link count* by exactly one?

a. create one file named `/1/2/3`
b. create a directory named `/1/2`
c. create a directory named `/1/2/3`
d. create one file named `/1/22`
e. create a directory named `/1/22`

21. In an empty directory, how many words are in file `c` after this:

```
    touch a b 1 b a ; ls >c
```

a. `0` *b.* `1` *c.* `4` *d.* `2` *e.* `3`

22. If directory `/a` contains these seven two-character names: `aa`, `ab`, `ac`, `ad`, `a?`, `a*`, `a.`, then which command removes *only* the single two-character name `a*` from the directory?

a. `rm /a/*` *b.* `rm /a/a*` *c.* `rm /a/a?`
d. `rm "/a/a*"` *e.* `rm /a*`

23. How many files are touched? `touch "1  " 2 3" "  ' '  4  5`

a. `4` *b.* `7` *c.* `6` *d.* `5` *e.* `3`

24. What is the link count of directory `d` after these successful commands?

```
    mkdir  d  d/a  d/b  d/c  d/c/z ; touch  d/x  d/y
```

a. `5` *b.* `6` *c.* `2` *d.* `4` *e.* `3`

25. What is the output on your screen after this:

```
    echo hi >a ; cp a b | wc -w
```

a. `3` *b.* `2` *c.* no output
d. `0` *e.* `1`

26. What is the link count of directory `dir` after these successful commands?

```
    mkdir dir ; cd dir ; touch one ; mkdir two
```

a. `5` *b.* `4` *c.* `2` *d.* `1` *e.* `3`

27. What is the link count of directory `dir` after these successful commands?

```
    mkdir dir ; cd dir ; touch a b c ; mkdir d e
```

a. `3` *b.* `5` *c.* `2` *d.* `7` *e.* `4`

28. Which command removes *only* this five-character name containing a special character: `date?`

a. `rm date/?` *b.* `rm date\\?` *c.* `rm ./date?`
d. `rm date\*` *e.* `rm ./date\?`

29. If you are in `/etc` and `ls -l` shows a symbolic link `bar -> ../foo` then dereference the absolute path of `bar` with no symbolic links:

a. `/foo` *b.* `/etc/foo/bar` *c.* `/etc/foo`
d. `/bar/foo` *e.* `/etc/bar/foo`

30. File `a` contains 2 lines. File `b` contains 3 lines. How many lines are in file `e` after this: `ln a d ; cp a f ; ln d c ; ln c e ; cat a b d f >e`

a. `3` *b.* `2` *c.* `5` *d.* `6` *e.* `9`

31. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 a
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 b
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 c
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 d
```

a. `3` *b.* `1` *c.* `0` *d.* `4` *e.* `2`

32. What is the link count of file `f` after these successful commands?
```
rm f ; touch f ; ln f bar
cp bar x ; ln x y ; ln bar z ; ln z a
```
a. `5` *b.* `2` *c.* `4` *d.* `1` *e.* `3`

33. If `/bin/prg` is a program that outputs `hi` and `/usr/bin/prg` is a program that outputs `foo` what is the output on your screen after this:
```
    PATH=/etc:/usr/bin:/bin ; prg
```
a. `hi` followed by `foo`
b. `bash: prg: command not found`
c. `foo` followed by `hi`
d. `foo`
e. `hi`

34. File `a` contains 3 lines. File `b` contains 4 lines. How many lines are output on your screen by this: `sort a | echo b`
a. `3` *b.* `3` followed by `4` *c.* `4`
d. `3` followed by `1` *e.* `1`

35. Which of these statements is true?
a. Only backslashes are strong enough to stop GLOB patterns from expanding.
b. Only double quotes are strong enough to stop GLOB patterns from expanding.
c. If `/y` is an empty directory, `echo /y/*` produces an error message.
d. Only single quotes are strong enough to stop GLOB patterns from expanding.
e. If `/x` is an empty directory, `sort /x/*` produces an error message.

36. If directory `/a` contains these seven two-character names: `aa`, `ab`, `ac`, `ad`, `a?`, `a*`, `a.`, then which command removes *only* the single two-character name `a*` from the directory?
a. `rm /a/a?` *b.* `rm /a/*` *c.* `rm /a*`
d. `rm /a/a*` *e.* `rm /a/a\*`

37. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:
```
111 -rw-r--r-- 2 me me 1 Jan 1 1:00 a
111 -rw-r--r-- 2 me me 1 Jan 1 1:00 b
222 -rw-r--r-- 3 me me 1 Jan 1 1:00 c
222 -rw-r--r-- 3 me me 1 Jan 1 1:00 d
```
a. `3` *b.* `2` *c.* `4` *d.* `0` *e.* `1`

38. In an empty directory, what is the output on your screen after this:
```
    echo hi >a ; ls | wc -w
```
a. `2` *b.* `0` *c.* `1`
d. `a` *e.* no output

39. How many arguments are passed to the command by the shell:
```
    <bar bar -b "-a" '-r' >bar bar bar
```
a. `6` *b.* `4` *c.* `5` *d.* `3` *e.* `7`

40. What is the output on your screen after these command lines:
```
echo 1 >x ; ln x y ; echo 2 >>y
head -1 x >y ; cat y
```
a. `1` *b.* `2` *c.* `1` followed by `2`
d. `2` followed by `1` *e.* no output

41. Which command removes *only* this four-character name containing a special character: `*xyz`
a. `rm ''*xyz''` *b.* `rm *"xyz"` *c.* `rm ''*xyz`
d. `rm *xyz` *e.* `rm "*xyz"`

42. What is the link count of file `f` after these successful commands?
```
rm f ; touch f ; ln f b ; cp f c
cp b x ; ln x y ; ln b z ; ln z a
```
a. `3` *b.* `5` *c.* `1` *d.* `2` *e.* `4`

43. File `a` contains 2 lines. File `b` contains 3 lines. How many lines are in file `c` after this: `ln a d ; ln b e ; cp d e >c`
a. `0` *b.* `4` *c.* `3` *d.* `2` *e.* `5`

44. What command will recursively find all pathnames named `foo` in `/bin`?
a. `ls -R 'foo' /bin`
b. `find /bin -name 'foo'`
c. `grep /bin -basename 'foo'`
d. `find foo -name '/bin'`
e. `grep 'foo' /bin`

45. File `a` contains 2 lines. File `b` contains 3 lines. How many lines are in file `c` after this: `ln a d ; ln d c ; cp c b ; sort a b d >c`
a. `6` *b.* `5` *c.* `0` *d.* `2` *e.* `4`

46. Which command shows the name of the current computer:
a. `history` *b.* `whois` *c.* `comname`
d. `hostname` *e.* `find`

47. How many arguments are passed to the command by the shell:
```
    <bat bat -b "-a -r" >bat bat bat
```
a. `4` *b.* `6` *c.* `7` *d.* `5` *e.* `3`

48. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:
```
111 -rw-r--r-- 1 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 2 me me 100 Jan 1 1:00 b
333 -rw-r--r-- 2 me me 100 Jan 1 1:00 c
444 -rw-r--r-- 1 me me 100 Jan 1 1:00 d
```
a. `1` *b.* `0` *c.* `3` *d.* `2` *e.* `4`

49. What is the link count of directory `d` after these successful commands?
```
    mkdir d ; mkdir d/a ; mkdir d/a/b ; mkdir d/a/c
```
a. `4` *b.* `2` *c.* `3` *d.* `5` *e.* `1`

50. What is the output on your screen after this:

```
echo one >x ; ln x y ; echo two >>y ; sort x
```

a. **two** followed by **one** *b.* **one**
c. **two** *d.* no output
e. **one** followed by **two**

51. In an empty directory, how many words are in file **a** after this:

```
echo It's redirected >b isn't it\? ; ls >a
```

a. **3** *b.* **0** *c.* **1** *d.* **4** *e.* **2**

52. File **a** contains 2 lines. File **b** contains 3 lines. How many lines are in file **c** after this: **sort a b >c ; cat a >>b ; cat c b >c a**

a. **12** *b.* **7** *c.* **0** *d.* **5** *e.* **8**

53. What does *quoting* mean on a shell command line?

a. setting the **PS1** variable to be your shell prompt
b. using more than one pathname argument to a command, e.g. **rm a b c**
c. using a leading tilde ("**~**") on a pathname to mean your **HOME** directory
d. typing a "control" character using the **[CTRL]** key
e. turning off the special meaning of shell meta-characters

54. In an empty directory, what is the output on your screen after this:

```
touch 1 2 3 ; cow="*" ; echo ""$cow""
```

a. **"1 2 3"** *b.* **"$cow"** *c.* **1 2 3**
d. ***** *e.* **$cow**

55. If I am in directory **/tmp** and **mt** is an empty sub-directory, what is true after this:

```
touch mt/bar ; mkdir mt/me ; cp mt/bar mt/../me
```

a. the directory **mt** now contains only a file named **me**
b. there is a second copy of the file **bar** in file **/tmp/me**
c. the directory **mt** is now empty
d. the command fails because the name **mt/../me** does not exist
e. there is a second copy of the file **bar** in directory **mt**

56. If you are in **/bin** and **ls -l** shows a symbolic link **foo -> /bar** then dereference the absolute path of **foo** with no symbolic links:

a. **/bin/bar** *b.* **/foo/bar** *c.* **/bin/foo/bar**
d. **/bar** *e.* **/bin/bar/foo**

57. What is the output on your screen after this:

```
mkdir foo ; rmdir foo | wc -w
```

a. **0** *b.* no output *c.* **2**
d. **1** *e.* **3**

58. If directory **dir** contains only these five two-character names: **a?**, **11**, **?1**, **1***, **.1**, then which command removes *only* the single two-character name **?1** from the directory?

a. **rm dir/?1** *b.* **rm dir/??** *c.* **rm dir/1***
d. **rm dir/*1** *e.* **rm dir/\\??**

59. In an empty directory, what is in file **out** after this:

```
ls nosuchfile | wc -w >out
```

a. **nosuchfile** *b.* nothing (empty file)
c. **1** *d.* **out**
e. **0**

60. File **a** contains 2 lines. File **b** contains 3 lines. How many lines are in file **c** after this: **ln a e ; ln b d ; ln d c ; cp d e ; sort a b e d >c**

a. **12** *b.* **7** *c.* **4** *d.* **6** *e.* **10**

61. What is the link count of directory **d** after these successful commands?

```
mkdir d ; cd d ; touch a ; mkdir b c
```

a. **6** *b.* **2** *c.* **4** *d.* **3** *e.* **5**

62. If you are in **/etc** and **ls -l** shows a symbolic link **bar -> foo** then dereference the absolute path of **bar** with no symbolic links:

a. **/etc/foo/bar** *b.* **/bar/foo** *c.* **/etc/bar/foo**
d. **/foo** *e.* **/etc/foo**

63. In an empty directory, what is the output on your screen after this:

```
touch 1 2 3 ; cow="*" ; echo "$cow"
```

a. **"$cow"** *b.* **1 2 3** *c.* **$cow**
d. **"1 2 3"** *e.* *****

64. Which command line shows the current date?

a. **date | bash** *b.* **bash date**
c. **bash <date** *d.* **bash >date ; cat date**
e. **echo date | bash**

65. What is in file **out** after this:

```
echo me >a ; ln a b ; echo hi >b ; ln a out ; rm a b
```

a. **hi** *b.* nothing (empty file)
c. **me** *d.* no such file (nonexistent)
e. **me** followed by **hi**

66. What is the link count of directory **d** after these successful commands?

```
mkdir d ; mkdir d/a ; touch d/b
```

a. **1** *b.* **2** *c.* **3** *d.* **4** *e.* **5**

67. What is true about this output from **ls -il foo bar**?

```
15 -rwxrwxrwx 2 bin bin 3 Jul 31 12:33 foo
15 -rwxrwxrwx 3 bin bin 3 Jul 31 12:33 bar
```

a. **foo** and **bar** are names for the same file
b. **foo** and **bar** each have three names (six names total)
c. **foo** and **bar** are two of three names for the same file
d. this output is not possible
e. **foo** and **bar** are names for different files

68. How many arguments are passed to the command by the shell:

```
    echo " 1 '2 3' 4 "5 6 ' 7 "8 ' >out
```

a. `6` *b.* `4` *c.* `2` *d.* `3` *e.* `5`

69. If directory `/a` contains these seven two-character names: `aa`, `ab`, `ac`, `ad`, `a*`, `a?`, `??`, then which command removes *only* the single two-character name `a?` from the directory?

a. `rm '/a/a?'` *b.* `rm /a/a?` *c.* `rm /a\?`
d. `rm "/a?"` *e.* `rm /a/?\?`

70. File `a` contains 2 lines. File `b` contains 3 lines. How many lines are in file `c` after this: `ln a d ; ln d c ; cat a b >c`

a. `4` *b.* `5` *c.* `0` *d.* `3` *e.* `2`

71. The option to `ls` that shows inode (index) numbers is:

a. `-l` *b.* `-a` *c.* `-i` *d.* `-x` *e.* `-1`

72. How many arguments are passed to the command by the shell:

```
    echo " 1 2 " three ' 4 ' five"6"
```

a. `1` *b.* `3` *c.* `4` *d.* `5` *e.* `9`

73. What is the link count of file `f` after these successful commands?

```
rm f ; touch f ; ln f bar
cp bar x ; ln x y ; ln bar z
```

a. `1` *b.* `3` *c.* `2` *d.* `4` *e.* `5`

74. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 1 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 1 me me 100 Jan 1 1:00 b
444 -rw-r--r-- 2 me me 100 Jan 1 1:00 c
444 -rw-r--r-- 2 me me 100 Jan 1 1:00 d
```

a. `2` *b.* `1` *c.* `0` *d.* `3` *e.* `4`

75. What is true about this output from `ls -il foo bar`

```
15 -r-x------ 2 me me 3 Jan 1 1:00 foo
99 -r-x------ 2 me me 3 Jan 1 1:00 bar
```

a. this output is not possible
b. `foo` and `bar` are names for the same file
c. `foo` and `bar` are names for different files
d. `foo` and `bar` each have three names (six names total)
e. `foo` and `bar` are two of three names for the same file

76. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 2 me me 100 Jan 1 1:00 b
333 -rw-r--r-- 2 me me 100 Jan 1 1:00 c
444 -rw-r--r-- 2 me me 100 Jan 1 1:00 d
```

a. `0` *b.* `1` *c.* `4` *d.* `3` *e.* `2`

77. File `a` contains 2 lines. File `b` contains 3 lines. How many lines are in file `c` after this: `ln a e ; ln b d ; ln d c ; cat e b >c`

a. `3` *b.* `4` *c.* `5` *d.* `0` *e.* `2`

78. How many arguments are passed to the command by the shell:

```
    <cow cow "-x "-y '-z' >cow cow
```

a. `5` *b.* `6` *c.* `3` *d.* `4` *e.* `7`

79. Dereference the following symlink `xyz` into its equivalent absolute path:

```
    ln -s ../../a/./b/../bar /tmp/a/b/xyz
```

a. `/tmp/bar` *b.* `/tmp/a/bar` *c.* `/tmp/b/xyz`
d. `/tmp/a/b/bar` *e.* `/tmp/b/bar`

80. If you are in `/bin` and `ls -l` shows a symbolic link `foo -> dir/bar` then dereference the absolute path of `foo` with no symbolic links:

a. `/foo/dir/bar` *b.* `/dir/bar`
c. `/bin/dir/bar/foo` *d.* `/bin/foo/dir/bar`
e. `/bin/dir/bar`

81. What is the link count of directory `dir` after these successful commands?

```
    mkdir dir ; touch foo ; cd dir ; ln ../foo bar
```

a. `2` *b.* `3` *c.* `4` *d.* `5` *e.* `1`

82. How many arguments are passed to the command by the shell:

```
    <f z " a 'b c' d " 1 2 ' g " h " ' >z
```

a. `4` *b.* `3` *c.* `5` *d.* `6` *e.* `2`

83. How many files are touched? `touch 1  "2 3"  ' 4 '  5`

a. `3` *b.* `7` *c.* `6` *d.* `5` *e.* `4`

84. What is in file `foo` after this:

```
    echo hi >a ; ln a b ; echo me >b ; ln a foo ; rm a b
```

a. no such file (nonexistent) *b.* `me`
c. `hi` *d.* nothing (empty file)
e. `hi` followed by `me`

85. What is the output of this in an empty directory:

```
    touch 1 13 .13 2 213 3 30 39 .31 ; echo [13]?
```

a. `1 13 3 30 39`
b. `13 30 39`
c. `[13]?`
d. an error message from `echo` saying `[13]?` does not exist
e. `13`

86. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 1 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 b
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 c
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 d
```

a. `1` *b.* `3` *c.* `4` *d.* `0` *e.* `2`

87. How many arguments are passed to the command by the shell:

```
    echo " one '2 three' 4 "five 6 ' 7 "8 ' >out
```

a. `5` *b.* `4` *c.* `2` *d.* `3` *e.* `6`

88. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 1 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 1 me me 100 Jan 1 1:00 b
333 -rw-r--r-- 1 me me 100 Jan 1 1:00 c
444 -rw-r--r-- 2 me me 100 Jan 1 1:00 d
```

a. `2` *b.* `4` *c.* `0` *d.* `3` *e.* `1`

89. If your `PATH` contained only the file names `/bin/sh`, `/bin/cat`, and `/bin/ls`, then what is the output on your screen of this command:

```
    cat /etc/passwd
```

a. `cat: bash: no such file or directory`
b. `bash: /bin/sh: command not found`
c. `cat: /etc/passwd: command not found`
d. `bash: /bin/cat: no such file or directory`
e. `bash: cat: command not found`

90. In an empty directory, what is the output on your screen after this:

```
    echo one >.bar ; echo .????*
```

a. an error message from `echo` saying `.????*` does not exist
b. `.bar`
c. `..  .bar`
d. `one`
e. `.????*`

91. What is the link count of file `foo` after these successful commands?

```
rm foo ; touch foo ; ln foo bar
cp bar x ; ln x y ; ln bar z ; ln z a
```

a. `4` *b.* `1` *c.* `5` *d.* `2` *e.* `3`

92. If your `PATH` variable contains `/bin:/usr/bin`, what is the output of this:

```
    echo '$PATH'
```

a. `'$PATH'`
b. `'/bin:/usr/bin'`
c. `$PATH`
d. `/bin:/usr/bin`
e. `echo: $PATH: No such file or directory`

93. What is the output on your screen after these command lines:

```
echo one >x ; ln x y ; echo ten >y
echo two >x ; cat y
```

a. `one` followed by `ten` and `two` *b.* no output on screen
c. `ten` *d.* `two`
e. `one`

94. The correct syntax to assign to a shell variable is:

a. `V = "foo bar"` *b.* `V = foo bar`
c. `V="foo bar"` *d.* `V=foo bar`
e. `"V=foo bar"`

95. In an empty directory, what is in file `count` after this:

```
    ls ??? | wc -w >count
```

a. `1 1 2` *b.* `1`
c. `0` *d.* `1 1 1`
e. nothing (empty file)

96. If you are in `/bin` and `ls -l` shows a symbolic link `bar -> ../dir/foo` then dereference the absolute path of `bar` with no symbolic links:

a. `/bin/bar/dir/foo` *b.* `/dir/foo`
c. `/bin/dir/foo/bar` *d.* `/bar/../dir/foo`
e. `/bin/dir/foo`

97. What is the output on your screen after this:

```
    PATH=/bin/cat:/bin/sh:/bin/ls ; ls nosuchfile
```

a. `bash: /bin/ls: command not found`
b. `ls: /bin/ls: command not found`
c. `bash: /bin/sh: No such file or directory`
d. `bash: ls: command not found`
e. `ls: nosuchfile: No such file or directory`

98. What is the link count of directory `a` after these successful commands?

```
    mkdir a ; mkdir a/b ; mkdir a/c ; mkdir a/b/c
```

a. `2` *b.* `1` *c.* `5` *d.* `4` *e.* `3`

99. Which command line always prints just the two characters `$x` on the screen?

a. `echo '$x'` *b.* `echo $$x` *c.* `echo "$x"`
d. `echo $x` *e.* `echo "$$x"`

100. What is in file `c` after this:

```
    echo B >b ; ln b a ; echo A >a ; ln a c ; rm a b
```

a. `A` *b.* no such file (nonexistent)
c. `B` *d.* `A` followed by `B`
e. nothing (empty file)

101. What is the link count of directory `dir` after these successful commands?

```
    mkdir dir ; mkdir dir/foo ; touch dir/bar
```

a. `5` *b.* `3` *c.* `2` *d.* `4` *e.* `1`

102. What is the link count of file `f` after these successful commands?

```
rm f ; touch f ; ln f b ; cp f g
cp b a ; ln a d ; ln b c ; cp c g
```

a. `1` *b.* `4` *c.* `3` *d.* `5` *e.* `2`

103. What is the output on your screen after these command lines:

```
echo one >x ; ln x y ; echo two >y
echo ten >x ; cat y
```

a. `one` followed by `two` and `ten` *b.* `ten`
c. `two` *d.* `one`
e. no output on screen

104. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 1 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 1 me me 100 Jan 1 1:00 b
333 -rw-r--r-- 1 me me 100 Jan 1 1:00 c
444 -rw-r--r-- 1 me me 100 Jan 1 1:00 d
```

a. `4` *b.* `1` *c.* `0` *d.* `3` *e.* `2`

105. How many arguments are passed to the command by the shell:

```
echo 'It's a bird! It's a plane!'
```

a. `1` *b.* `3` *c.* `2` *d.* `5` *e.* `4`

106. If `/bin/foo` is a program that outputs `mom` and `/usr/bin/foo` is a program that outputs `dad`, what is the output on your screen after this:

```
PATH=/dev:/usr/bin:/usr:/bin:/etc ; /bin/foo
```

a. `mom`
b. `bash: /bin/foo: command not found`
c. `dad` followed by `mom`
d. `mom` followed by `dad`
e. `dad`

107. If `/bin/foo` is a program that outputs `one` and `/usr/bin/foo` is a program that outputs `two`, what is the output on your screen after this:

```
PATH=/bin/ls:/home:/usr/bin/cat:/etc ; foo
```

a. `bash: foo: command not found`
b. `two`
c. `two` followed by `one`
d. `one`
e. `one` followed by `two`

108. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 a
222 -rw-r--r-- 1 me me 100 Jan 1 1:00 b
333 -rw-r--r-- 2 me me 100 Jan 1 1:00 c
333 -rw-r--r-- 2 me me 100 Jan 1 1:00 d
```

a. `1` *b.* `4` *c.* `3` *d.* `2` *e.* `0`

109. In an empty directory, what is the output on your screen after this:

```
echo one >.bar ; ls .????*
```

a. `one`
b. an error message from `ls` saying `.????*` does not exist
c. `..  .bar`
d. `.bar`
e. `.????*`

110. What command will recursively show disk usage in directories?

a. `find` *b.* `df` *c.* `tree` *d.* `du` *e.* `ls`

111. What is the link count of file `a` after these successful commands?

```
ln a d ; cp a f ; ln d c ; ln f g ; ln c e
```

a. `1` *b.* `5` *c.* `4` *d.* `2` *e.* `3`

112. How many files are touched? `touch 1  "2 3"  ' '  4  5`

a. `5` *b.* `4` *c.* `3` *d.* `6` *e.* `7`

113. What is in file `c` after this:

```
echo foo >a ; ln a b ; echo bar >>b ; ln a c ; rm a
```

a. no such file (nonexistent) *b.* `foo`
c. `foo` followed by `bar` *d.* nothing (empty file)
e. `bar`

114. If `/bin/foo` is a program that outputs `one` and `/usr/bin/foo` is a program that outputs `two`, what is the output on your screen after this:

```
PATH=/etc:/usr/bin:/usr:/bin:/dev ; foo
```

a. `two`
b. `bash: foo: command not found`
c. `two` followed by `one`
d. `one`
e. `one` followed by `two`

115. If `/bin/pig` is a program that outputs `xx` and `/usr/bin/pig` is a program that outputs `foo` what is the output on your screen after this:

```
PATH=/home:/bin:/dev:/usr/bin ; pig
```

a. `bash: pig: command not found`
b. `xx` followed by `foo`
c. `foo`
d. `xx`
e. `foo` followed by `xx`

116. Which command line allows programs in the current directory to execute without preceding the names with `./`? (P.S. Security Risk! Don't do this!)

a. `PATH=/usr/bin/.:$HOME` *b.* `$PATH=.:$HOME:/usr/bin`
c. `PATH=./$HOME:/usr/bin` *d.* `PATH=/bin:/usr/bin:.`
e. `$PATH=/usr/bin:./bin`

117. What is the link count of file **foo** after these successful commands?

```
rm foo ; touch foo ; ln foo bar
cp bar a ; ln a b ; ln bar c ; cp c a
```

a. **3** *b.* **5** *c.* **4** *d.* **2** *e.* **1**

118. File **a** contains 2 lines. File **b** contains 3 lines. How many lines are output on your screen by this: `cat b | cat a`

a. **3** followed by **2** *b.* **5** *c.* **3**
d. **2** followed by **3** *e.* **2**

119. How many arguments are passed to the command by the shell:

```
echo 'It's "1 2" isn't it? I can't decide.
```

a. **6** *b.* **5** *c.* **2** *d.* **4** *e.* **3**

120. In an empty directory, what is the output on your screen after this:

```
echo hi >foo ; cp foo bar | wc -w
```

a. **3** *b.* **1** *c.* **2**
d. **0** *e.* no output

121. In an empty directory, what is the output on your screen after this:

```
echo hi >a ; mv a b ; ln b c ; ls >wc -l
```

a. **2** *b.* no output *c.* **a**
d. **0** *e.* **1**

122. What is usually in the environment variable **$HOME**?

a. the relative path of the system **/home** directory
b. the relative path of your login home directory
c. the absolute path of the system **/home** directory
d. the relative path of the ROOT directory
e. the absolute path of your login home directory

123. If the file **bat** contained the word **foo**, what is the output on your screen after this: `PATH=/etc/passwd:/bin/ls:/bin/cat ; /bin/ls bat`

a. no output on screen
b. **bash: /bin/ls: command not found**
c. **foo**
d. **bat**
e. **/bin/ls: bat: No such file or directory**

124. What is the link count of directory **x** after these successful commands?

```
mkdir x ; mkdir x/y ; mkdir x/z ; mkdir x/y/z
```

a. **3** *b.* **2** *c.* **5** *d.* **4** *e.* **1**

125. What is the link count of file **foo** after these successful commands?

```
rm foo ; touch foo ; ln foo bar ; ln bar x
cp bar a ; ln a b ; ln x c ; cp c d
```

a. **1** *b.* **2** *c.* **4** *d.* **5** *e.* **3**

126. What is the link count of an empty directory?

a. **1** *b.* **2** *c.* **4** *d.* **3** *e.* **0**

127. In an empty directory, how many words are in file **c** after this:

```
touch a ; mv b a >b ; ls >c
```

a. **2** *b.* **4** *c.* **0** *d.* **3** *e.* **1**

128. What is true about this output from `ls -il foo bar`

```
35 -rw-rw-r-- 2 me me 3 Jan 1 1:00 foo
36 -rw-rw-r-- 2 me me 3 Jan 1 1:00 bar
```

a. **foo** and **bar** each have two names (four names total)
b. **foo** and **bar** are names for the same file
c. **foo** and **bar** each have three names (six names total)
d. this output is not possible
e. **foo** and **bar** are two of three names for this file

129. If you want a user-defined alias in all your **bash** shells, what do you do?

a. put the alias into the **/bin/bash** file for next log in
b. create the alias and then type **save** to save it to all shells
c. put the alias into the **/etc/passwd** file for next log in
d. define the alias in my file **$HOME/.bashrc**
e. put the alias into the **/etc/group** file for next log in

130. What is usually in the environment variable **$PATH**?

a. a colon-separated list of your **passwd** file fields
b. the absolute path of your login shell
c. a colon-separated list of directories containing command names
d. the absolute path of your login home directory
e. the absolute path of the system **/path** directory

131. What is the link count of directory **foo** after these successful commands?

```
mkdir foo ; cd foo ; touch a b c
```

a. **2** *b.* **5** *c.* **3** *d.* **1** *e.* **4**

132. What is the link count of directory **d** after these successful commands?

```
mkdir d ; mkdir d/a d/b ; touch d/c d/e
```

a. **3** *b.* **1** *c.* **5** *d.* **4** *e.* **2**

133. Rewrite as a simplified absolute path:

```
/../../var/./a/../../var/b/../../etc/./bar/../foo
```

a. **/etc/bar/foo** *b.* **/var/foo** *c.* **/var/b/foo**
d. **/var/a/foo** *e.* **/etc/foo**

134. File **a** contains 2 lines. File **b** contains 3 lines. How many lines are in file **c** after this: `ln a d ; ln d e ; ln b f >c`

a. **5** *b.* **0** *c.* **3** *d.* **4** *e.* **2**

135. How do you execute the program **foo** in the current directory?

a. **foo/** *b.* **$HOME/foo** *c.* **./foo**
d. **foo/.** *e.* **/foo**

136. What is the link count of file `f` after these successful commands?

```
    cp f x ; ln f a ; ln x y ; ln a z ; ln a b
```

a. `2` *b.* `6` *c.* `5` *d.* `3` *e.* `4`

137. What is the output on your screen after this:

```
    echo 1 >x ; ln x y ; echo 2 >>y ; sort x
```

a. `1` followed by `2` *b.* no output *c.* `2` followed by `1`
d. `1` *e.* `2`

138. If `/bin/xxx` is a program that outputs `one` and `/usr/bin/xxx` is a program that outputs `two`, what is the output on your screen after this:

```
    PATH=/etc:/usr/bin:/usr:/bin ; /bin/xxx
```

a. `two` followed by `one`
b. `two`
c. `bash: /bin/xxx: command not found`
d. `one`
e. `one` followed by `two`

139. What is the link count of file `f` after these successful commands?

```
rm f ; touch f ; ln f bar
cp bar x ; ln x y ; ln y z
```

a. `0` *b.* `1` *c.* `3` *d.* `2` *e.* `4`

140. In an empty directory, what is the output on your screen after this:

```
    touch A a ; echo * ">*"
```

a. `A a >*` *b.* `A a` *c.* `A a >A a`
d. No output *e.* `* >*`

141. Rewrite as a simplified absolute path:

```
/usr/./bin/../lib/../../etc/../usr/./lib/../bin/./bar
```

a. `/usr/lib/bar` *b.* `/usr/bin/bar` *c.* `/etc/bar`
d. `/usr/bar` *e.* `/bar`

142. How many arguments are passed to the command by the shell:

```
    <foo foo -x " " -z -r" " >foo 'foo foo'
```

a. `8` *b.* `9` *c.* `5` *d.* `7` *e.* `6`

143. Rewrite as a simplified absolute path:

```
/home/me/../you/../../etc/../home/me/../you/../me/../foo
```

a. `/home/foo` *b.* `/foo`
c. `/etc/foo` *d.* `/home/me/foo`
e. `/home/you/foo`

144. In an empty directory, what is the output on your screen after this:

```
    touch A a ; echo * >"*" ; ls
```

a. `* >*` *b.* No output *c.* `A a >*`
d. `* A a` *e.* `A a >A a`

145. How many arguments are passed to the command by the shell:

```
    <cow cow "-x" -y '-z' >cow cow
```

a. `4` *b.* `5` *c.* `6` *d.* `3` *e.* `2`

146. If `mt` is an empty sub-directory, what is true after this:

```
    touch mt/bar ; mkdir bar ; mv mt/bar mt/../bar/me
```

a. the `mkdir` fails because `bar` already exists
b. the directory `mt` is now empty
c. the directory `mt` now contains only a file named `me`
d. there is a second copy of the file `bar` in the file named `me`
e. the command fails because the name `mt/../bar/me` does not exist

147. What is the link count of directory `z` after these successful commands?

```
    mkdir z ; cd z ; touch a ; ln a b ; ln a c
```

a. `4` *b.* `3` *c.* `5` *d.* `1` *e.* `2`

148. Which command removes *only* this four-character name containing a special character: `?xyz`

a. `rm '?xyz'` *b.* `rm ?xyz` *c.* `rm ''?xyz''`
d. `rm ?'xyz'` *e.* `rm ''?xyz`

149. In an empty directory, what is in file `foo` after this:

```
    echo hi >foo ; ls nosuchfile | cat >foo
```

a. nothing (empty file)
b. `foo`
c. `ls: cannot access nosuchfile`
d. `hi`
e. `nosuchfile`

150. What is the link count of file `f` after these successful commands?

```
rm f ; touch f ; ln f a ; ln a b
cp f c ; ln c x ; rm b ; mv a b
```

a. `3` *b.* `1` *c.* `2` *d.* `0` *e.* `4`

151. What is the link count of file `foo` after these successful commands?

```
rm foo ; touch foo ; ln foo bar
cp bar x ; ln x y ; ln y z
```

a. `4` *b.* `3` *c.* `1` *d.* `2` *e.* `0`

152. Which of the following `PATH` statements makes the most sense?

a. `PATH=/bin:/usr/bin:/etc/passwd`
b. `PATH=/bin:/etc/passwd:/usr/bin`
c. `PATH=/bin/ls:/etc/passwd:/usr/bin`
d. `PATH=/bin:/usr/bin`
e. `PATH=/bin/bash:/usr/bin:/bin`

153. What is the link count of directory `d` after these successful commands?

```
    mkdir d ; touch f ; cd d ; ln ../f x
```

a. `2` *b.* `1` *c.* `3` *d.* `5` *e.* `4`

154. How many arguments are passed to the command by the shell:

```
    echo " 1 2 "three ' 4 ' five"6"
```

a. `1` *b.* `9` *c.* `4` *d.* `5` *e.* `3`

155. What displays on your screen given this command:

```
date >date ; pwd >pwd ; head date | tail pwd
```

a. `tail` reads the pipe and the `pwd` and displays both together
b. only the `date` displays because `tail` ignores the pipe
c. nothing displays because `tail` ignores the pipe
d. only the `pwd` displays because `tail` ignores the pipe
e. `head` displays the `date` and `tail` displays the `pwd`

156. How many arguments are passed to the command by the shell:

```
echo 'And it's not hard, it's just logical.'
```

a. `3` *b.* `4` *c.* `5` *d.* `7` *e.* `6`

157. Which of the following is true, given this long directory listing:

```
drwxr-x--x 128 me me 32 Jan 1 1:00 dir
```

a. The number 32 is the count of links (names) this directory has.
b. The number 32 is the size of this directory.
c. The number 128 is the size of this directory.
d. The number 128 is the inode number of this directory.
e. The number 32 is the inode number of this directory.

158. How many arguments are passed to the command by the shell:

```
<foo foo " a 'b c' d " e f ' g " h " ' >foo
```

a. `5` *b.* `4` *c.* `2` *d.* `3` *e.* `6`

159. What is the output on your screen of this unquoted command line:

```
mkdir a ; touch b a/b1 a/b2 ; find a -name b*
```

a. `a/b1 a/b2` *b.* `b1 b2` *c.* `b`
d. `b a/b1 a/b2` *e.* no output

160. You enter this `cp a/b c/`
and get `cp: a: No such file or directory`
because:

a. you forgot to specify the destination file name after `c/`
b. the command `cp` is not in your search PATH
c. directory `a` does not exist
d. directory `c` does not exist
e. pathname `a` exists but is a file, not a directory

161. What is the output of this in an empty directory:

```
date >.date ; users >.users ; echo .?*
```

a. an error message from `echo` saying `.?*` does not exist
b. `.date`
c. `..  .date  .users`
d. `.date  .users`
e. `.?*`

162. Which command line below shows only lines 6-10 of file `foo`?

a. `head -6 foo | tail -10` *b.* `tail -15 foo | head -5`
c. `tail -10 foo | head -6` *d.* `head -10 foo | tail -5`
e. `head -10 foo | tail -6`

163. What is the output on your screen after this: `echo hi >out | wc -w`

a. `1` *b.* `0` *c.* `2`
d. no output *e.* `3`

164. If `mt` is an empty sub-directory, what is true after this:

```
touch bar ; mkdir foo ; mv mt/../bar mt/foo
```

a. the directory `foo` now contains a file named `bar`
b. the directory `mt` now contains a file named `bar`
c. the command fails because `mt/foo` is not a directory
d. the directory `mt` is still empty
e. the directory `mt` now contains a file named `foo`

165. How many files are touched? `touch '1  "2 3  '4'" '5`

a. `1` *b.* `4` *c.* `3` *d.* `5` *e.* `2`

166. Which command line would show the index (inode) number of a file?

a. `find -i file` *b.* `cat -i file` *c.* `cat -l file`
d. `ls -i file` *e.* `ls -l file`

167. What is in file `c` after this:

```
echo A >a ; ln a b ; echo B >b ; ln a c ; rm a b
```

a. `A` followed by `B` *b.* no such file (nonexistent)
c. `A` *d.* nothing (empty file)
e. `B`

168. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 a
111 -rw-r--r-- 2 me me 100 Jan 1 1:00 b
222 -rw-r--r-- 2 me me 100 Jan 1 1:00 c
222 -rw-r--r-- 2 me me 100 Jan 1 1:00 d
```

a. `0` *b.* `2` *c.* `1` *d.* `3` *e.* `4`

169. Which command line shows just the count of words in the file?

a. `wc file | awk '[print #2]'`
b. `wc file | awk '{print 2}'`
c. `wc file | awk '[print $2]'`
d. `wc file | awk '{print $2}'`
e. `wc file | awk '{print #2}'`

170. To change to the parent directory, do this:

a. `cd` *b.* `pwd ..` *c.* `cd .`
d. `pwd` *e.* `cd ..`

171. What is the link count of file **f** after these successful commands?

```
rm f ; touch f ; cp f x
ln f a ; ln x y ; ln a z ; ln x b
```

a. **3** *b.* **6** *c.* **2** *d.* **4** *e.* **5**

172. How many arguments are passed to the command by the shell:

```
    echo ' one two ' three ' four ' 5'6'
```

a. **5** *b.* **1** *c.* **4** *d.* **6** *e.* **9**

173. What is the link count of directory **d** after these successful commands?

```
    mkdir d ; cd d ; touch f ; ln f a ; ln f b
```

a. **1** *b.* **5** *c.* **4** *d.* **2** *e.* **3**

174. Which command line makes pathnames **/usr/local/bin** and **/usr/bin** lead to the same directory?

a. **ln . /usr/local** *b.* **mkdir /usr/local**
c. **touch /usr/local** *d.* **ln -s . /usr/local**
e. **rmdir /usr/local**

175. What is true about this output from **ls -il foo bar**

```
15 -r-x-----x 2 me me 3 Jan 1 1:00 foo
15 -r-x-----x 2 me me 3 Jan 1 1:00 bar
```

a. **foo** and **bar** are names for different files
b. **foo** and **bar** are names for the same file
c. this output is not possible
d. **foo** and **bar** are two of three names for this file
e. **foo** and **bar** each have three names (six names total)

176. How many arguments are passed to the command by the shell:

```
    <bar bar -b"-a '-r' >bar" bar >out
```

a. **3** *b.* **5** *c.* **6** *d.* **4** *e.* **2**

177. If **mt** is an empty sub-directory, what is true after this:

```
    touch foo ; mkdir bar ; mv foo bar/mt
```

a. the directory **bar** now contains a file named **foo**
b. the command fails because **bar/mt** is not a directory
c. the directory **mt** now contains a directory named **bar**
d. the directory **mt** now contains a file named **foo**
e. the directory **mt** is still empty

178. File **a** contains 2 lines. File **b** contains 3 lines. How many lines are in file **d** (not in **c**) after this:

```
ln a d ; ln d c ; ln c e ; cat a a b b c c d d e e >c
```

a. **18** *b.* **2** *c.* **10** *d.* **21** *e.* **6**

179. Which option to **ls** displays the directory itself and not its contents?

a. **-a** *b.* **-i** *c.* **-l** *d.* **-d** *e.* **-R**

180. If file **one** occupies one disk block, how many disk blocks are in use after this:

```
    cp one foo ; ln foo two ; ln two bar ; ln one cow
```

a. **2** *b.* **4** *c.* **1** *d.* **3** *e.* **5**

181. What is true about this output from **ls -il foo bar**

```
15 -r-x------ 2 me me 3 Jan 1 1:00 foo
15 -rwxrwxrwx 2 me me 3 Jan 1 1:00 bar
```

a. this output is not possible
b. **foo** and **bar** are names for different files
c. **foo** and **bar** are names for the same file
d. **foo** and **bar** each have three names (six names total)
e. **foo** and **bar** are two of three names for the same file

182. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 3 me me 100 Jan 1 1:00 a
111 -rw-r--r-- 3 me me 100 Jan 1 1:00 b
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 c
222 -rw-r--r-- 3 me me 100 Jan 1 1:00 d
```

a. **2** *b.* **4** *c.* **0** *d.* **3** *e.* **1**

183. If file **foo** occupies one disk block, how many disk blocks are in use after this:

```
    cp foo bar ; ln bar one ; cp one two ; ln one pig
```

a. **1** *b.* **3** *c.* **4** *d.* **2** *e.* **5**

184. If **/bin/foo** is a program that outputs **one** and **/usr/bin/foo** is a program that outputs **two**, what is the output on your screen after this:

```
    PATH=/dev:/usr/bin:/usr:/bin:/etc ; /bin/foo
```

a. **bash: /bin/foo: command not found**
b. **two**
c. **one** followed by **two**
d. **one**
e. **two** followed by **one**

185. Which command line outputs inode/filename pairs for names in the current directory, sorted by inode number?

a. **sort -n | ls -ai** *b.* **ls -i * > sort -n**
c. **ls -node * > sort -n** *d.* **ls ./* | sort -node**
e. **ls -ai | sort -n**

186. What displays on your screen given this command:

```
    ls >ls ; wc ls >wc ; sort ls | cat wc
```

a. nothing displays because **cat** ignores the pipe
b. **sort** displays the **ls** and **cat** displays the **wc**
c. **cat** reads the pipe and the **wc** and displays both together
d. only the **ls** displays because **cat** ignores the pipe
e. only the **wc** displays because **cat** ignores the pipe

187. How many arguments are passed to the command by the shell:

```
    <pig pig -x " " -z -r" " >pig pig pig
```

a. **9** *b.* **5** *c.* **8** *d.* **6** *e.* **7**

188. In an empty directory, what is the output on your screen after this:

```
echo one >.bar ; echo .??*
```

a. an error message from `echo` saying `.??*` does not exist
b. `.??*`
c. `one`
d. `..  .bar`
e. `.bar`

189. How many arguments are passed to the command by the shell:

```
<foo foo " a 'b c' d " e ' f " g " ' >foo
```

a. `4` *b.* `5` *c.* `2` *d.* `6` *e.* `3`

190. If `/bin/foo` is a program that outputs `hi` and `/usr/bin/foo` is a program that outputs `mom` what is the output on your screen after this:

```
PATH=/etc:/usr/bin:/bin ; foo
```

a. `hi` followed by `mom`
b. `mom`
c. `bash: foo: command not found`
d. `mom` followed by `hi`
e. `hi`

191. If your terminal type is `xterm`, what is the output of this: `echo '$TERM'`

a. `xterm` *b.* `$TERM`
c. `'xterm'` *d.* `'$TERM'`
e. no output on screen

192. What is the link count of file `f` after these successful commands?

```
rm f ; touch f ; cp f x
ln f a ; ln x y ; ln a z ; ln z q
```

a. `5` *b.* `6` *c.* `2` *d.* `3` *e.* `4`

193. In an empty directory, what is the output on your screen after this:

```
touch a ; ls | wc -w
```

a. no output *b.* `3` *c.* `1`
d. `2` *e.* `0`

194. In an empty directory, what is the output on your screen after this:

```
echo one >.bar ; echo .?*
```

a. `.bar`
b. `one`
c. an error message from `echo` saying `.?*` does not exist
d. `.?*`
e. `..  .bar`

195. In an empty directory, what is the output on your screen after this:

```
touch 1 2 3 ; cow="*" ; echo '$cow'
```

a. `1 2 3` *b.* `$cow` *c.* `'1 2 3'`
d. `'$cow'` *e.* `*`

196. What is the resulting link count of empty directory `dir` after these successful commands? `cd dir ; touch foo ; ln foo one ; ln foo two`

a. `3` *b.* `1` *c.* `2` *d.* `5` *e.* `4`

197. Which of the following is true, given this long directory listing:

```
drwxr-x--x 128 me me 32 Jan 1 1:00 dir
```

a. The number 32 is the count of links (names) this directory has.
b. The number 128 is the size of this directory.
c. The number 32 is the inode number of this directory.
d. The number 128 is the inode number of this directory.
e. The number 128 is the count of links (names) this directory has.

198. If I have a directory named `a/b`, which action would increase its *link count* by exactly one?

a. create a hard link to directory `b` named `b2`
b. create a directory named `a/b2`
c. create a directory named `a/b/c`
d. create a file named `a/b2`
e. create a file named `a/b/c`

199. If `/bin/bat` is a program that outputs `foo` and `/usr/bin/bat` is a program that outputs `hi` what is the output on your screen after this:

```
PATH=/usr:/usr/bin:/bin ; bat
```

a. `bash: bat: command not found`
b. `hi` followed by `foo`
c. `foo` followed by `hi`
d. `hi`
e. `foo`

200. Which command usually goes in your `.bash_profile` file?

a. `.bash_profile source` *b.* `.bashrc source`
c. `cat .bashrc` *d.* `source ./.bashrc`
e. `source ./.bash_profile`

201. How many files are touched? `touch "1"  2 3  " "  4  5`

a. `6` *b.* `4` *c.* `3` *d.* `5` *e.* `7`

202. If `/bin/pig` is a program that outputs `hi` and `/usr/bin/pig` is a program that outputs `foo` what is the output on your screen after this:

```
PATH=/etc:/usr/bin:/bin ; pig
```

a. `foo` followed by `hi`
b. `hi`
c. `bash: pig: command not found`
d. `foo`
e. `hi` followed by `foo`

203. How many files are touched? `touch 1  "2 3  ' 4 '"  5`

a. `4` *b.* `2` *c.* `1` *d.* `5` *e.* `3`

204. What is the link count of directory `z` after these successful commands?

```
    mkdir z ; cd z ; touch a b ; mkdir c d e
```

a. `3` *b.* `6` *c.* `7` *d.* `5` *e.* `4`

205. If `/bin/bat` is a program that outputs `foo` and `/usr/bin/bat` is a program that outputs `bar` what is the output on your screen after this:

```
    PATH=/usr:/usr/bin:/bin ; bat
```

a. `foo`
b. `bash: bat: command not found`
c. `bar` followed by `foo`
d. `bar`
e. `foo` followed by `bar`

206. Create a symbolic link under `/usr` named `bar` that has target `xy`:

a. `ln -s 'xy' '/usr/bar'`
b. `ln -s /usr/bar 'xy'`
c. `ln -s '/usr/xy' /usr/bar`
d. `ln -s /usr/bar '/usr/xy'`
e. `ln -s 'xy' /bar/usr`

207. If files occupy one disk block, how many disk blocks will the system free up if I remove these four file names:

```
111 -rw-r--r-- 1 me me 1 Jan 1 1:00 a
222 -rw-r--r-- 1 me me 1 Jan 1 1:00 b
333 -rw-r--r-- 1 me me 1 Jan 1 1:00 c
444 -rw-r--r-- 2 me me 1 Jan 1 1:00 d
```

a. `1` *b.* `2` *c.* `0` *d.* `4` *e.* `3`

208. If `/bin/xxx` is a program that outputs `one` and `/usr/bin/xxx` is a program that outputs `two`, what is the output on your screen after this:

```
    PATH=/usr:/usr/bin:/etc:/bin ; xxx
```

a. `bash: xxx: command not found`
b. `two` followed by `one`
c. `one`
d. `one` followed by `two`
e. `two`

209. What is the link count of directory `dir` after these successful commands?

```
    mkdir dir ; cd dir ; touch foo ; mkdir a b c
```

a. `3` *b.* `2` *c.* `5` *d.* `1` *e.* `4`

210. How many arguments are passed to the command by the shell:

```
    <foo foo " a 'b c' d " e ' f " g " ' >foo h
```

a. `6` *b.* `5` *c.* `2` *d.* `3` *e.* `4`

211. A "dangling symlink" is a symlink to:

a. a non-existent target *b.* a directory
c. a special device file *d.* a parent directory
e. the current directory

212. In an empty directory, what is the output on your screen after this:

```
    echo one >.bar ; echo .*
```

a. `.*`
b. `.   ..   .bar`
c. `one`
d. an error message from `echo` saying `.*` does not exist
e. `.bar`

213. What is the link count of directory `z` after these successful commands?

```
    mkdir z ; mkdir z/a ; touch z/b z/c z/d
```

a. `4` *b.* `2` *c.* `1` *d.* `5` *e.* `3`

214. What is the link count of directory `d` after these successful commands?

```
    mkdir d ; mkdir d/a ; mkdir d/b ; mkdir d/b/c
```

a. `5` *b.* `1` *c.* `3` *d.* `4` *e.* `2`

215. If `/bin/xxx` is a program that outputs `one` and `/usr/bin/xxx` is a program that outputs `two`, what is the output on your screen after this:

```
    PATH=/bin/xxx:/usr/bin/xxx:/etc/passwd ; xxx
```

a. `bash: xxx: command not found`
b. `one` followed by `two`
c. `two` followed by `one`
d. `two`
e. `one`

216. If the file `pig` contained the word `bar`, what is the output on your screen after this: `PATH=/etc/passwd:/bin/ls:/bin/who ; /bin/cat pig`

a. `/bin/cat: pig: No such file or directory`
b. no output on screen
c. `bash: /bin/cat: command not found`
d. `pig`
e. `bar`

217. What is usually in the environment variable `$SHELL`?

a. the relative path of the system `/shell` directory
b. the relative path of your login shell
c. the relative path of the `/home/shell` directory
d. the absolute path of the system `/shell` directory
e. the absolute path of your login shell

218. **Did you read all the words of the test instructions on page one?**

a. `Sim` *(Yes - Portuguese)* *b.* `Tak` *(Yes - Polish)*
c. `Taip` *(Yes - Lithuanian)* *d.* `Igen` *(Yes - Hungarian)*
e. `Jes` *(Yes - Esperanto)*