Evaluation: 39 Questions **Name:** _____

Important Instructions

1. Read all instructions and both sides of all pages.
2. Manage your time when answering questions on this test. Answer the questions you know, first.

(Office use only: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39)

1. If **foo** is a script containing the line **TERM=linux ; export TERM**, what is the output of the following sequence of **bash** commands:

```
TERM=vt100 ; ./foo ; echo "$TERM"
```

† **a**. **vt100**
b. **linux**
c. **foo**
d. **TERM**
e. **$TERM**

2. If file **bar** contains the line **a=abc** then what is the **bash** output of this sequence of three commands:

```
a=123 ; source bar ; echo "I see '$a' here."
```

† **a**. **I see 'abc' here.**
b. **I see '123' here.**
c. **I see '$a' here.**
d. **I see $a here.**
e. **"I see abc here."**

3. What is the output of the following sequence of **bash** commands:

```
cd /bin && echo "cd $(pwd)"
```

† **a**. **cd /bin**
b. no output
c. **cd 0pwd)**
d. **cd $(pwd)**
e. **bash: cd: /bin: No such file or directory**

4. In an empty directory, how many files will be created using the following **bash** shell two-command sequence:

```
yy='aa  b  cc  d' ; touch $yy
```

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

5. In an empty directory, how many files will be created using the following **bash** shell two-command sequence:

```
zzz='1111  2222  3333' ; touch "$zzz"
```

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

6. In an empty directory, what is the shell output of these three commands:

```
touch xx .x xy .y xz .z ; a='x* y*' ; echo "$a"
```

† **a**. **x* y***
b. **xx xy**
c. **xx xy xz y***
d. **$a**
e. ***x *y**

7. In an empty directory, what is the shell output of these three commands:

```
touch .1 .2 .3 11 12 ; b='.1* .2*' ; echo '$b'
```

† **a**. **$b**
b. **.1* .2***
c. **'.1* .2*'**
d. **.1 .2**
e. **11 .1 12 .2**

8. In an empty directory, what is the length of the longest file name created by the following **bash** shell two-command sequence:

```
ok='1 12 123 1234' ; touch '$ok'
```

† **a**. 3 characters
b. 4 characters
c. 2 characters
d. 1 character
e. 13 characters

9. What is the output of the following sequence of **bash** commands:

```
false && echo "linux     rocks $?"
```

† **a**. no output
b. **linux rocks 1**
c. **linux   rocks 0**
d. **linux   rocks 1**
e. **linux rocks 0**

10. What is the **bash** shell output of this command sequence:
```
true && echo space     junk $?
```
† **a**. **space junk 0**
b. **space junk ?**
c. **space junk ?**
d. **space junk 1**
e. no output

11. How many arguments are passed to the command by the shell on this command line: **<cow cow "-x" -y '-z' >cow cow**
† **a**. **4**
b. **5**
c. **2**
d. **3**
e. **6**

12. A shell script named **foo** is executed as follows:
```
./foo  aa  "bb  cc" '  dd'  ee
```
Inside the script is the line: **argv.sh "$@"**
What is the count of arguments that the **argv.sh** command will display?
† **a**. **4**
b. **3**
c. **2**
d. **5**
e. **6**

13. A shell script named **foo** is executed as follows:
```
./foo  11  '22  22  22'  "33  33"
```
Inside the script is the line: **argv.sh $@**
What is the count of arguments that the **argv.sh** command will display?
† **a**. **6**
b. **5**
c. **4**
d. **3**
e. **2**

14. A shell script named **foo** is executed as follows:
```
./foo  11  22  "33  44"  55
```
Inside the script is the line: **argv.sh "$*"**
What is the count of arguments that the **argv.sh** command will display?
† **a**. **1**
b. **2**
c. **3**
d. **4**
e. **5**

15. If **dog=12** and **cat=99** then which of the following **bash** command lines outputs only the word **hi** (and nothing else)?
† **a**. **[dog = dog] && echo hi**
b. **[dog -ne cat] && echo hi**
c. **[!dog = cat] && echo hi**
d. **[dog -ne cat] || echo hi**
e. **[dog!=dog] || echo hi**

16. What is the output of the following sequence of **bash** commands:
```
x=1 ; y=2 ; test $x -le $y ; echo $?
```
† **a**. **0**
b. **1**
c. the number 0 or 1 followed by another 0 or 1 on a new line
d. **test: $x: integer expression expected**
e. no output

17. What is the output of the following sequence of **bash** commands:
```
x=cow ; y=dog ; test -z $x ; echo $?
```
† **a**. **1**
b. **0**
c. the number 0 or 1 followed by another 0 or 1 on a new line
d. **test: $x: integer expression expected**
e. no output

18. If **x=cow** and **y=dog** then what is the output of the following sequence of **bash** commands: **[$x = dog -o $y = cow] ; echo $?**
† **a**. **1**
b. **0**
c. the number 0 or 1 followed by another 0 or 1 on a new line
d. **test: $x: integer expression expected**
e. no output

19. If **x=cow** and **y=dog** then what is the output of the following sequence of **bash** commands: **[$x = cow -a $y = cow] ; echo $?**
† **a**. **1**
b. **0**
c. the number 0 or 1 followed by another 0 or 1 on a new line
d. **test: $x: integer expression expected**
e. no output

20. If **x=pig** and **y=dog** then what is the output of the following sequence of **bash** commands: **if $x = $y ; then echo $y ; fi**

† **a**. **bash: pig: command not found**
b. **test: pig: integer expression expected**
c. **test: $x: integer expression expected**
d. **dog**
e. no output

21. If a **bash** shell script named **foo** contains the line:
if ["$1" = '$2'] ; then echo SAME ; fi
then which of the following command lines will produce **SAME** as output?

† **a**. **./foo '$2' bar**
b. **./foo bar bar**
c. **./foo "bar" 'bar'**
d. **./foo "$1" '$2'**
e. **./foo $2 $2**

22. Which **bash** command sequence correctly compares the two numbers and prints **OK**?

† **a**. **if [4 -gt 3] ; then echo OK ; fi**
b. **if [4 -gr 3] ; then echo OK ; fi**
c. **if [4 > 3] ; then echo OK ; fi**
d. **if [! 4 <= 3] ; then echo OK ; fi**
e. **if (! 4 < 3) ; then echo OK ; fi**

23. If **cow=5** and **dog=5**, which **bash** command sequence correctly compares the two numbers as equal and prints **OK**?

† **a**. **if test $cow -eq $dog ; then echo OK ; fi**
b. **if test cow -eq dog ; then echo OK ; fi**
c. **if [cow = dog] ; then echo OK ; fi**
d. **if (cow == dog) ; then echo OK ; fi**
e. **if [$cow==$dog] ; then echo OK ; fi**

24. Which **bash** command sequence correctly searches for the **string** and then prints **OK** if it is found inside the password file?

† **a**. **if grep string /etc/passwd ; then echo OK ; fi**
b. **if [grep string /etc/passwd] ; then echo OK ; fi**
c. **if test string /etc/passwd ; then echo OK ; fi**
d. **if test string = /etc/passwd ; then echo OK ; fi**
e. **if [test string /etc/passwd] ; then echo OK ; fi**

25. If variable **cow** might contain nothing (a null value - defined but empty), which **bash** command sequence correctly tests for this and prints **OK**?

† **a**. **if ["" = "$cow"] ; then echo OK ; fi**
b. **if [$cow -eq :] ; then echo OK ; fi**
c. **if [$cow -eq ""] ; then echo OK ; fi**
d. **if [''$cow'' = ''''] ; then echo OK ; fi**
e. **if ["$cow" = *] ; then echo OK ; fi**

26. Which **bash** command sequence below always outputs just the word **OK** only if the first argument is either readable or executable?

† **a**. **if [-r "$1" -o -x "$1"]; then echo OK;fi**
b. **if ["-r $1" || "-x $1"]; then echo OK;fi**
c. **if ["$1" -eq -r -o "$1" -eq -x]; then echo OK;fi**
d. **if [-r -o -x "$1"]; then echo OK;fi**
e. **if [-r || -x "$1"]; then echo OK;fi**

27. Which line below is most likely to be the beginning of an error message?

† **a**. **echo 1>&2 "... "**
b. **echo 1<&2 "... "**
c. **echo 2>&1 "... "**
d. **echo 2<$1 "... "**
e. **echo 2>$1 "... "**

28. Which line below puts the count of the number of lines in the password file into the variable **foo**?

† **a**. **foo=$(wc -l </etc/passwd)**
b. **foo=$(cat -c /etc/passwd)**
c. **foo=[wc /etc/passwd | echo $1]**
d. **foo=[cat -l /etc/passwd]**
e. **foo=[grep -c /etc/passwd]**

29. What is the output of the following sequence of **bash** commands:
echo wc >wc ; wc wc >wc ; sort wc

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

30. Which line below passes three *separate* arguments to the **sort** command when placed inside a shell script named **foo** invoked by the command line:
    ./foo 111 222 333

† **a**. **sort "$@"**
b. **sort "$*"**
c. **sort "$#"**
d. **sort "$1 $2 $3"**
e. **sort "$? $? $?"**

31. Given the following **bash** shell command line: **read xx yy zz** which user keyboard input line below will assign the text **22** to the shell variable named **yy**?

† **a**. **11 22 33**
b. **xx=11 yy=22 zz=33**
c. **11,22,33**
d. **11:22:33**
e. **11;22;33**

32. What is the **bash** shell output of this two-command sequence if run in a directory containing 888 files with names that are all the numbers from **1** to **888** inclusive: **cow="*" ; echo '$cow'**

† **a**. **$cow**
b. *****
c. **'$cow'**
d. the file names **1** through **888**
e. the file names **1** through **888**, surrounded by quotes

33. What is the **bash** shell output of this two-command sequence if run in a directory containing 123 files with names that are all the numbers from **1** to **123** inclusive: **bat="*" ; echo "$bat"**

† **a**. *****
b. **$bat**
c. **"$bat"**
d. the file names **1** through **123**
e. the file names **1** through **123**, surrounded by quotes

34. What is the **bash** shell output of this two-command sequence if run in a directory containing 765 files with names that are all the numbers from **1** to **765** inclusive: **foo="*" ; echo $foo**

† **a**. the file names **1** through **765**
b. all the file names that start with an asterisk ('*****')
c. an asterisk ('*****') and the file names **1** through **765**
d. *****
e. **$foo**

35. Which **bash** command line below allows programs in the current directory to execute without preceding the names with **./**?

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

36. A shell script named **foo** is executed as follows:
    ./foo 1 "2 3 4" 5
    Inside the script is the line: **echo "$2"**
    What is the output from this line?

† **a**. **2 3 4**
b. **2**
c. **"2**
d. **$2**
e. a bash error message: unbound (undefined) variable

37. Select the correct **bash** shell order of command line processing:

† **a**. aliases, redirection, variables, globs
b. aliases, variables, redirection, globs
c. aliases, variables, globs, redirection
d. aliases, globs, variables, redirection
e. redirection, aliases, globs, variables

38. If these two lines are put in an executable script named **foo**:
    #!/bin/mv dog
    echo cow
    What is the result of the command line: **./foo**

† **a**. The file **dog** is renamed to be **./foo**
b. The file **dog** is copied to the file **./foo**
c. The word "cow" appears on the screen
d. The **mv** command displays an error message about a missing argument
e. The file **dog** appears on the screen followed by the word "hi"

39. What is the output of the following sequence of **bash** commands:
    wc='one two' ; test wc = wc

† **a**. no output
b. **1 2 8 wc**
c. **1**
d. **0**
e. **test: too many arguments**

Answer Key - DAT 2330 – Ian Allen – Winter 2003 - DAT 2330 Test #3b - Unix Final - 15%

Office use only: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

```
  1. a                          Count of a:  39  100%
  2. a
  3. a                          With 5 choices: 39
  4. a                            1 2 3 4 5 6 7 8 9 10 11
    5. a                          12 13 14 15 16 17 18 19
    6. a                          20 21 22 23 24 25 26 27
    7. a                          28 29 30 31 32 33 34 35
    8. a                          36 37 38 39
  9. a
 10. a                          Macro .cmd splits: 20
 11. a                          Macro .ans splits: 0
 12. a
   13. a
   14. a
   15. a
   16. a
 17. a
 18. a
 19. a
 20. a
   21. a
   22. a
   23. a
   24. a
 25. a
 26. a
 27. a
 28. a
   29. a
   30. a
   31. a
   32. a
 33. a
 34. a
 35. a
 36. a
   37. a
   38. a
   39. a
```