**Linux– Shell Programming MCQ**

1. What will be output of following command:

$ **echo** "The process id is" $$$$

a**)** The process **id** is $$

b**)** The process **id** is $**<**pid**>**$**<**pid**>**

c**)** The process **id** is **<**pid**><**pid**>**

d**)** The process **id** is $$$$

View Answer

Answer: c  
Explanation: None.

2. What would be the current working directory at the end of the following command sequence?

$ **pwd**

**/**home**/**user1**/**proj

$ **cd** src

$ **cd** generic

$ **cd** .

$ **pwd**

a)/home/user1/proj  
b)/home/user1/proj/src  
c)/home/user1  
d)/home/user1/proj/src/generic  
View Answer

Answer:d d  
Explanation: None.

3. How do you print the lines between 5 and 10, both inclusive  
a) cat filename | head | tail -6  
b) cat filename | head | tail -5  
c) cat filename | tail +5 | head  
d) cat filename | tail -5 | head -10

Answer: a

4. Create a new file “new.txt” that is a concatenation of “file1.txt” and “file2.txt”  
a) cp file.txt file2.txt new.txt  
b) cat file1.txt file2.txt > new.txt  
c) mv file[12].txt new.txt  
d) ls file1.txt file2.txt | new.txt

Answer: b

5. which of these is NOT a valid variable in bash  
a) \_\_ (double underscore)  
b) \_1var (underscore 1 var )  
c) \_var\_ (underscore var underscore)  
d) some-var (some hyphen var)

Answer: d

6. What is the output of the following code:

os=Unix

**echo** 1.$os 2."$os" 3.'$os' 4.$os

a)1.Unix2.Unix3.Unix4.Unix  
b)1.Unix2.Unix3.$os4.Unix  
c)1.Unix2.Unix3.Unix4.$os  
d) 1.Unix 2.$os 3.$os 4.$os

Answer :b

7. What is the return value ($?) of this code:

os = Unix

**[**$osName = UnixName**]** **&&** **exit** 2

**[**${os}Name = UnixName**]** **&&** **exit** 3

a)0  
b)1  
c)2  
d) 3

Answer :d

8. What is the output of the following program?

x = 3; y = 5; z = 10;

**if** **[(** $x -eq 3 **)** -a **(** $y -eq 5 -o $z -eq 10 **)]**

**then**

**echo** $x

**else**

**echo** $y

**fi**

a)1  
b)3  
c)5  
d) Error

Answer: b

9. What is the output of the following program?

**[** -n $HOME **]**

**echo** $?

**[** -z $HOME **]**

**echo** $?

a**)** 0

1

b**)** 1

0

c**)** 0

0

d**)** 1

1

Answer: a

10. What is the output of the following program?

b =

**[** -n $b **]**

**echo** $?

**[** -z $b **]**

**echo** $?

a**)** 1

1

b**)** 2

2

c**)** 0

0

d**)** 0

1

Answer:c

11.The expression expr -9 % 2 evaluates to:  
a)0  
b)1  
c)-1  
d) 2

Answer: c

12. The statement z = ‘expr 5 / 2’ would store which of the following values in z?  
a)0  
b)1  
c)2  
d)2.5

Answer: c

13. When the return value of any function is not specified within the function, what function returns?  
a)nothing  
b)exit status of the last command executed  
c) 0  
d) none of the mentioned  
View Answer

Answer:a b  
Explanation: None.

14. Parameters can be passed to a function  
a) by using the parameter variables $1, $2, $3…….  
b) by using the environment variables  
c) by using the parameter & environment variables  
d) none of the mentioned  
View Answer

Answer:a a  
Explanation: None.

15. Which of the following command provides the list of the functions defined in the login session?  
a) declare -f  
b) declare -F  
c) both declare -f and -F  
d) none of the mentioned  
View Answer

Answer:a c  
Explanation:’declare -F’ provides just the name of the functions and ‘declare -f’ provides their definitions also.

16. The keyword ‘local’ is used  
a) to define a variable within a function for its local scope  
b) to redefine any global variable  
c) this is not a valid keyword  
d) none of the mentioned  
View Answer

Answer:c a  
Explanation: None.

17. Functions improves the shell’s programmability significantly, because  
a) when we invoke a function, it is already in the shell’s memory, therefore a function runs faster than seperate scripts  
b) function will not provides a piece of code for repetative tasks  
c) all of the mentioned  
d) none of the mentioned  
View Answer

Answer: a  
Explanation: None.

18. What is the output of this program?

1. *#!/bin/sh*
2. var="Sanfoundry"
3. san\_function**()** **{**
4. var="Linux"
5. **echo** $var
6. **}**
7. san\_function
8. **exit** 0

a) Sanfoundry  
b) Linux  
c) Command not found  
d) None of the mentioned  
View Answer

Answer: b  
Explanation: If local variable name is same as the global variable, it overlays the variable, but only within the function.

19. What is the output of this program?

1. *#!/bin/sh*
2. san\_function**()** **{**
3. **echo** "Welcome to the Sanfoundry"
4. **printf** "World of Linux**\n**"
5. **}**
6. **unset** -f san\_function
7. san\_function
8. **exit** 0

a) Welcome to the Sanfoundry  
b) World of Linux  
c) both Welcome to the Sanfoundry and World of Linux  
d) nothing will print  
View Answer

Answer: d  
Explanation: Function definition was deleted before calling the function. command ‘unset -f function\_name’ deletes the function definition.

20. What is the output of this program?

1. *#!/bin/bash*
2. **function** san\_function1 **{**
3. **echo** "This is first function"
4. **}**
5. san\_function2**()** **{**
6. **echo** "This is second function"
7. **}**
8. san\_function1
9. san\_function2
10. **exit** 0

a) This is the first function  
b) This is the second function  
c) This is the first function  
This is the second function  
d) program will generate error because first function definition is not correct  
View Answer

Answer: c  
Explanation: In bash shell, functions can be defined in both the ways, used in the script.  
This is first function  
This is second function  
root@ubuntu:/home/sanfoundry#

21. What is the output of this program?

1. *#!/bin/sh*
2. **echo** "Just call the function"
3. san\_function
4. san\_function**()** **{**
5. **echo** "This is a function"
6. **}**
7. **exit** 0

a) only first string will print without any error  
b) only second string will print without any error  
c) both strings will print  
d) none of the mentioned  
View Answer

Answer: d  
Explanation: Function must be defined prior to call. Hence only first string will print and program will generate an error also.

22. What is the output of this program?

1. *#!/bin/sh*
2. san\_function1**()** **{**
3. a=5
4. **echo** "This is the first function"
5. san\_function2
6. **}**
7. san\_function2**()** **{**
8. **echo** "This is the second function"
9. san\_function3
10. **}**
11. san\_function3**()** **{**
12. **echo** "This is the third function"
13. **}**
14. san\_function1
15. **exit** 0

a) This is the first function  
This is the second function  
This is the third function  
b) This is the first function  
This is the third function  
This is the second function  
c) This is the second function  
This is the first function  
This is the third function  
d) This is the third function  
This is the first function  
This is the second function  
View Answer

Answer: a  
Explanation: None.