



Bash Programming

TOTAL POINTS 10

1. Which of the following are requirements for variable names?

1 point

- ☒ Every character in the name is lowercase.
- ☒ The variable name starts with a letter.
- ☐ Every character in the name is uppercase.
- ☐ Numbers are not allowed in variable names.

2. What does the exit status of a program indicate?

1 point

- ☐ The exit status of a program indicates how a program will be completed once it's exhausted all of its code.
- ☒ The exit status of a program indicates whether the program was executed successfully or whether an error occurred.
- ☐ The exit status of a program indicates the value that was computed by the program.
- ☐ The exit status of a program indicates how many programs were running at the same time as a particular program.

3. What is printed to the console by the following command?

1 point

```
1 echo Demetrius || [[ 6 -eq 7 ]] || echo Helena && echo Hermia || [[ 7 -gt 4 ]]
```

☒

```
1 Demetrius
2 Hermia
```

☐

```
1 Demetrius
2 Helena
```

☐

```
1 Helena
2 Hermia
```

☐ Nothing is printed to the console.

4. Consider the following program called numrange.sh:

1 point

```
1 #!/usr/bin/env bash
2 # File: numrange.sh
3
4 odd=$(echo "$1 % 2" | bc)
5
6 if [[ $odd -eq 0 ]]
7 then
8     status="even"
9 else
10    status="odd"
11 fi
12
13 if [[ $1 -gt 0 ]] && [[ $1 -lt 10 ]]
14 then
15     location="in"
16 else
17     location="out of"
18 fi
19
20 echo "This number is $status and $location range."
21
```

Which of the following is the result of commands below?

```
1 bash numrange.sh 6
2 bash numrange.sh 11
3 bash numrange.sh 400 10
```

☒

```
1 This number is even and in range.
2 This number is odd and out of range.
3 This number is even and out of range.
```

- ☐

```
1 This number is even and out of range.
2 This number is odd and in range.
3 This number is even and out of range.
```
- ☐

```
1 This number is odd and out of range.
2 This number is even and in range.
3 This number is even and out of range.
```
- ☐

```
1 This number is even and in range.
2 This number is odd and out of range.
3 - error - too many arguments
```

5. What is the result of the script below?

1 point

```
1 lab=(jeff roger brian)
2 lab[3]=sean
3 lab=("${lab[*]}") "${lab[*]}"
4 echo ${#lab[*]}
```

- ☐ 1
- ☒ 2
- ☐ 9
- ☐ 6

6. Consider the following program called repseq.sh:

1 point

```
1 #!/usr/bin/env bash
2 # File: repseq.sh
3
4 sequence=$(eval echo ${1..$2})
5
6 for i in $sequence
7 do
8   compute=$(echo "$i % 3" | bc)
9   result="$result $compute"
10 done
11
12 echo $result
13
```

Which of the commands below would create the following output?

```
1 1 2 0 1 2 0 1 2 0
```

- ☐

```
1 bash repseq.sh 1 6 3
```
- ☒

```
1 bash repseq.sh 1 9 3
```
- ☐

```
1 bash repseq.sh 1 9 2
```
- ☐

```
1 bash repseq.sh 1 6 2
```

7. What's the purpose of the **local** keyword?

1 point

- ☐ The **local** keyword allows you to create a function such that the function can be used within your shell the same way you would use a command.
- ☐ The **local** keyword stores the value of several variables locally so that they can be accessed later on within a script.

- ☒ The **local** keyword allows you to assign the value of a variable within a function without changing the global value of that variable.
- ☐ The **local** keyword ensures that all of the actions taken by a particular function do not affect the global computing environment.

8. Which of the following are **not** part of the Unix Philosophy?

1 point

- ☒ Programs should be composable.
- ☒ Programs should have easy to understand error messages.
- ☒ A program should do one thing well.
- ☒ Programs should be quiet.
- ☒ A program should run quickly.

9. What actions are taken by the following commands?

1 point

1	chmod a+x my_program	
2	chmod go-rw my_program	

- ☐ 1. Allows anyone to execute **my_program**.
2. Prevents the owner from reading or modifying **my_program**.
- ☐ 1. Allows only the owner to execute **my_program**.
2. Prevents the owner from reading or modifying **my_program**.
- ☒ 1. Allows anyone to execute **my_program**.
2. Prevents anyone other than the owner from reading or modifying **my_program**.
- ☐ 1. Allows only the owner to execute **my_program**.
2. Prevents anyone other than the owner from reading or modifying **my_program**.

10. What is one reason you might want to modify the **PATH** environmental variable?

1 point

- ☐ You can make functions available to you on the command line by including the PATH variable inside of the definition of a function.
- ☐ Modifying the PATH makes it easier to switch between programs when you are using multiple shells at once.
- ☒ You can add a directory containing your own programs to the PATH which allows you to access them on the command line.
- ☐ The PATH can be modified in the bash profile which is where aliases are defined. The bash profile is run every time you start a shell.

☒ I, **Piyush Sambhi**, understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

[Learn more about Coursera's Honor Code](#)



Save

Submit