These questions practice arguments. You should begin by looking at the files beneath the samples/args directory beneath the class public work area.

1. What does the following command do:
   ```
   set -- "Hello there" "Lucy" ""
   ```
2. After executing the above command,
   - what variable contains the string *Hello there*?
   - what is the value of $#? What about $*
3. Next, execute `shift 1`. What are the values of $# and $1 now?
4. Xensin executes the following commands. What is the output?
   ```
   set -- $(date)
   echo "$1 $4"
   ```
You have written a shell script `showargs.bash` to display the positional parameters. For each command using `showargs.bash` below, indicate how many positional parameters `showargs.bash` would see and what each of their values is. The value of the variable `name` is *Jim Jones*.

   Example:
   ```
   showargs.bash a b c
   ```
   you would write:
   ```
   Three positional parameters. $1=a $2=b $3=c
   ```
   (There is a version of `showargs.bash` in the samples directory if you'd like to try these out.)
5. `showargs.bash` *His name is "'$name'"*
6. `showargs.bash"hello\tthere" $name`
7. `showargs.bash "\$name" '\$name' \$name`
8. `showargs.bash check#thisout check #thisout`

You have executed the following commands:
```
unset Name Address
Name="Amy Lynn"
Addr="124 Easy St."
export Name
```
You have a shell script `mkaddr.bash`. It has the single line
```
echo -e "\$Name\n\$Addr\n\$1"
```
9. What is the output of the command `mkaddr.bash San Francisco`

The following practice should be done using files beneath the samples directory (beneath the public work area for the class)

10. Practice with arguments using the shell script `args.bash` in the `args` subdirectory to practice your quoting and shell expansion rules. Define some variables and give various combinations of them with and without different kinds of quotes to `args.bash` as arguments. Predict the results before you run the program to test yourself.

   **Solutions on the next page**
Solutions

1. sets the positional parameters ($1...$9,$#,$*) of the current shell

2.
   • $1
   • 3

3. $# is 2 and $1 is Lucy

4. The day of the week followed by a blank followed by the time hh:mm:ss

5. Four positional parameters. $1=His $2=name $3=is $4='Jim Jones'
   (the single quotes are part of the variable’s value.)

6. Three positional parameters. $1=hello\tthere $2=Jim $3=Jones

7. Three positional parameters. $1=$name $2=\$name $3=$name

8. Two positional parameters. $1=check#thisout $2=check
   (# introduces a comment when it is at the start of the line or there is a space before it, so the first #
   is not a comment, but the second is.)

9. The output is below. Since only the Name variable was exported, mkaddr.bash did not get a copy
   of the Addr variable, and it output an empty line. Further, since San Francisco was not quoted,
   $1 was just San:

   Amy Lynn

   San