1. **Why are functions advantageous to have in your programs?**

Ans. Functions will help us to not write the code again and again and we can call it from any part of the program i.e function enables reusability of code.

2. **W+ `hen does the code in a function run: when it's specified or when it's called?**

Ans. The code in the function runs when it is called.

Eg.

```
def add(a,b):
    return a+b
```

```
add(4,3)
```

```
7
```

3. **What statement creates a function?**

Ans. We have to write the keyword “def” before the function name to create a function.

4. **What is the difference between a function and a function call?**

Ans. A function is a statement that we write and a function call denotes to get the function in use. Eg.

```
def add(a,b):  #function
    return a+b
```

```
add(4,3)   #function call
```

```
7
```

5. **How many global scopes are there in a Python program? How many local scopes?**

Ans. There is only one global and local scope. In case of global scope the variable is declared outside the function and it can be accessed anywhere within the function or outside the function. In case of local scope the variable declared inside the function would be only available within the function and cannot be used outside the block.Eg:

```
x=10  #global scope
```

```
def square():
    print(x**2)
```

```
square()
```

```
100
```

```
def square1():
    a = 2       #local scope
    print(a**2)
```

```
square1()
```

```
4
```

```
b = 5
c = b + a
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-9-cf036de9b3b8> in <module>
      1 b = 5
----> 2 c = b + a

NameError: name 'a' is not defined
```

6. What happens to variables in a local scope when the function call returns?

Ans. When the function call returns the memory allocated for the local variables are deallocated.

7. What is the concept of a return value? Is it possible to have a return value in an expression?

Ans. Return value is used instead of print since the type of print will give us NoneType however the return will give us the data type which the variable inside the function returns. This will help us to perform various operations using return which is not possible if we use the keyword “print”.

Yes return value can be part of an expression inside a function. Eg.

```
def sub():
    a = 7
    b = 6
    return a-b
```

```
sub()
```

```
1
```

8. If a function does not have a return statement, what is the return value of a call to that function?

Ans. If a function do not have a return statement, it will return NoneType.

```
def sub():
    a = 7
    b = 6
    a - b
```

```
type(sub())
```

```
NoneType
```

9. How do you make a function variable refer to the global variable?

Ans.

```
a=10 #gobal variable
```

```
def div():
    b = a       #b is a local variable and refers to global variable
    c = 20/b
    return c
```

```
div()
```

```
2.0
```

10. What is the data type of None?

Ans. Data type is NoneType

```
type(None)
```

```
NoneType
```

11. What does the sentence import areallyourpetsnamederic do?

Ans. This will import a module name areallyourpetsnamederic from python

12. If you had a bacon() feature in a spam module, what would you call it after importing spam?

Ans. Import spam

spam.bacon()

13. What can you do to save a programme from crashing if it encounters an error?

Ans. If the line of code encounters an error we can use exception handling in this case.

Whichever line of code is causing error we have to keep it under try clause.

14. What is the purpose of the try clause? What is the purpose of the except clause?

Ans. Try clause is used to prevent the program from crashing if there is an error in the line of code. The except clause will show us the error or the line of statement we intend to be shown when the line of code in try clause fails or it shows error.Eg.

```
try:
    a = 10
    b = 0
    c = a/b
except Exception as e:
    print(e)
    print("this is the except clause which shows the error")
```

```
division by zero
this is the except clause which shows the error
```