Assignmet2

April 24, 2023

```
[ ]: 1.What are the two values of the Boolean data type? How do you write them?
```

ANS:-In Python, the two values of the Boolean data type are represented by the keywords ”True” and ”False”.This keywords are case-sensitive, so ”True” and ”False” must be capitalized. you can assign Boolean values to a variable:

x = True y = False

```
[ ]: 2. What are the three different types of Boolean operators?
```

ANS:- Three types of Boolen operators are:- 1.”and” operator: This operator returns ”True” if both the operands are true; otherwise, it returns ”False”.

2. ”or” operator: This operator returns ”True” if at least one of the operands is true; otherwise, it returns ”False”.

3.”not” operator: This operator returns the opposite of the Boolean value of the operand. If the operand is true, it returns ”False”, and if the operand is false, it returns ”True”.

```
[ ]: 3. Make a list of each Boolean operator's truth tables
     (i.e. every possible combination of Boolean values for the operator and what it␣
      ↪evaluate ).
```

The truth tables for each of the three Boolean operators: 1.”and” operator: Operand 1 Operand 2 Result 0 1 0 1 0 0 1 1 1 0 0 0

2.”or” operator:- Operand 1 Operand 2 Result 0 1 1 1 0 1 1 1 1 0 0 0

3.”not” operator:- Operand Result 0 1
1 0

```
[ ]: 4. What are the values of the following expressions?
     (5 > 4) and (3 == 5) ==> False
     not (5 > 4) ==> False
     (5 > 4) or (3 == 5) ==> True
     not ((5 > 4) or (3 == 5)) ==> False
     (True and True) and (True == False) ==> False
     (not False) or (not True) ==> Ture
```

```
[ ]: 5. What are the six comparison operators?
```

list of the six comparison operators in Python: 1. ”==” (equal to) 2. ”!=” (not equal to) 3. ”>” (greater than) 4. ”>=” (greater than or equal to) 5. ”<” (less than) 6. ”<=” (less than or equal to)

[ ]:
```
6. How do you tell the difference between the equal to and assignment operators?
Describe a condition and when you would use one.
```

The equal to operator “==” is used to compare two values to see if they are equal, whereas the assignment operator “=” is used to assign a value to a variable.

x = 5 # this is an assignment statement, where the value 5 is assigned to the variable x y = 10

if x == y: # this is a comparison statement that checks if x is equal to y print(“x is equal to y”)

[ ]:
```
7. Identify the three blocks in this code:
spam = 0
if spam == 10:
print('eggs')
if spam > 5:
print('bacon')
else:
print('ham')
print('spam')
print('spam')
```

ANS:- ham spam spam

[ ]:
```
8. Write code that prints Hello if 1 is stored in spam, prints Howdy if 2 is␣
  ↪stored in spam,
and prints Greetings! if anything else is stored in spam.
```

spam=int(input(“enter the number:”)) if spam == 1: print(“Hello”) elif spam == 2: print(“Howdy”) else: print(“Greetings!”)

[ ]:
```
9.If your programme is stuck in an endless loop, what keys you’ll press?
```

ANS:- Ctrl + C

[ ]:
```
10. How can you tell the difference between break and continue?
```

ANS:-break and continue are both control statements used in loops to alter the flow of execution. break is used to exit the loop prematurely and immediately move on to the next statement after the loop. When the break statement is executed within a loop, the loop is terminated, and the program continues to execute from the next statement after the loop. continue, on the other hand, is used to skip the current iteration of the loop and move on to the next iteration. When the continue statement is executed within a loop, the program jumps immediately to the next iteration of the loop.

[ ]:
```
11. In a for loop, what is the difference between range(10), range(0, 10), and␣
  ↪range(0, 10, 1)?
```

In a for loop, range(10) means default strating value will be 0 and generates a sequence of numbers from 0 to 9 (inclusive) with a step of 1. This is equivalent to range(0, 10, 1). range(0, 10) generates a sequence of numbers from 0 to 9 (inclusive) with a step of 1. This is equivalent to range(10). range(0, 10, 1) generates a sequence of numbers from 0 to 9 (inclusive) with a step of 1. This is equivalent to range(10).

```
[ ]: 12. Write a short program that prints the numbers 1 to 10 using a for loop.
     Then write an equivalent program that prints the numbers 1 to 10 using a while␣
       ↪loop.
```

#using for loop for i in range(1,11): print(i)

#using While loop: i = 1 while i <= 10: print(i) i += 1

```
[ ]: 13. If you had a function named bacon() inside a module named spam, how would␣
       ↪you call it after importing spam?
```

ANS:- import spam

spam.bacon()

```
[ ]:
```