Operators in Python

Very Short Answer Type Questions (1 marks each)

Question 1:
Write the output of the given Python code :

```
a = 0
a+ =2
print (a)
```

Answer:
2

Question 2:
Write the output of the given Python code :

```
print (3 + 4)
print (3 – 4)
print (3 * 4) print (3/4)
print (3 % 2)
print (3**4) #3 to the fourth power
print (3 // 4) #floor division
```

Answer:

```
7
-1
12
0.75
1
81
0
```

Question 3:
Write the output of the given Python code :

```
a = 20
if a > = 22:
print(“if”)
elif a >= 21:
print(“elif”)
```

else:
print(“else”)

Answer:

else

Question 4:
What is the another name of Comparison Operators?
Answer:
Relational Operators.

Question 5:
Which Operator is used for assigning a value into a variable?
Answer:
Assignment Operators.

Question 6:
Which Operator is used for comparison of values?
Answer:
Logical Operators.

Question 7:
What is the use of “+” operator?
Answer:
The “+” operator adds values on either side of the operator.

Question 8:
What is the use of operator?
Answer:
The operator subtracts right hand operand from left hand operand.

Question 9:
What is the use of “*” operator?
Answer:
The “*” operator multiplies values on either side of the operator.

Question 10:
What is the use of operator?
Answer:
The 7” operator divides left hand operand by right hand operand.

Question 11:
What is the use of “%” operator?
Answer:
The “%” operator divides left hand operand by right hand operand and returns remainder.

Question 12:
What is the use of “**” operator?
Answer:
The “**” operator performs exponential (power) calculation on operators.

Question 13:
What is the use of 7/” operator?
Answer:
The 7/” operator results is the quotient in which the digits after the decimal point are removed.

Question 14:
What is the use of “= =” operator?
Answer:
It checks if the value of two operands are equal or not, if yes then condition becomes true.

Question 15:
What is the use of “! =” operator?
Answer:
It checks if the value of two operands are equal or not, if values are not equal then condition becomes true.

Question 16:
Why do we use ” < > “operator?
Answer:
It checks if the value of two operands are equal or not, if values are not equal then condition becomes true.

Question 17:
Describe the use of “>” operator?
Answer:
It checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.

Question 18:
Why we use “<” operator?

Answer:
The “<” operator checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.

Question 19:
What value has been check by ” > = ” operator?
Answer:
It checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.

Question 20:
Write syntax for “greater than or equal to” operator.
Answer:
a > = b

Question 21:
What value have been check by ” < = ” operator?
Answer:
It checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.

Question 22:
Is ” < =” a Assignment operator ?
Answer:
No,”< =” is not assignment operator.

Question 23:
Optimize these statements as a Python programmer.

```
1 word = ‘word’
2 print word._len_()
```

Answer:

```
1 word = ‘word’
2 print len(word)
```

Question 24:
How many types of operations are supported by the Python language ?
Answer:
Python language supports the following types of operations.

• Arithmetic operations
• Comparison (i.e. Rotational) operations
• Assignment operations
• Logical operations
• Bitwise operations
• Membership operations
• Identity operations.

Question 25:
What will be the output of the following code :

```
1 a = 1
2 a, b = a+1, a+1
3 print a
4 print b
```

Answer:

```
2
2
```

Question 26:
Is there an equivalent of C's "?:" ternary operator in Python ?
Answer:

No.

Short Answer Type Questions (2 marks each)

Question 1:
What do you mean by Operator precedence?
Answer:
Operator precedence determines the grouping of terms in an expression. This affects how an expression is evaluated. Certain operators have higher precedence than others; for example, the multiplication operator has higher precedence than the addition operator.

Question 2:
Assume if A = 60; and B = 13; then find the values of the following :
A&B
A|B

Answer:
Here in binary format they will be as follows:
A = 0011 1100
B = 0000 1101
hence
A&B = 0000 1100 and A|B = 00111101

Question 3:
Assume if A = 60; and B = 13, then find the values of the following.
A ^ Band ~A
Answer:
Here in binary format they will be as follows :
A = 00111100 B = 0000 1101
hence
A ^ B = 0011 0001 and ~A = 1100 0011

Question 4:
Explain assigning values to variables in Python.
Answer:
Python variables do not have to be explicitly declared to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

The operand to the left of the = operator is the name of the variable, and the operand to the right of the = operator is the value stored in the variable.
For example:

```
# !/usr/bin/python
counter = 100 # An integer assignment
miles = 1000.0 # A floating point
name = “John” # A string
print counter
print miles
print name
```

while running this program, this will produce the following result:

```
100
1000.0
John
```

Question 5:
Write the output of the following Python code

```
a = 6
b = 7
c = 42
print 1, a = = 6
print 2, a = = 7
print 3, a = = 6 and b = = 7
print 4, a = = 7 and b = = 7
print 5, not a = = 7 and b = = 7
print 6,a==7orb==7
print 7, a==7orb==6
print 8, not (a = = 7 and b = = 6)
print 9, not a = = 7 and b = = 6
```

Answer:

1 True
2 False
3 True
4 False
5 True
6 True
7 False
8 True
9 False

Question 6:
What do you mean by expression “a = 5” ?
Answer:
This statement assigns the integer value 5 to the variable a. The part at the left of the assignment operator (=) is known as the 1 value (left value) and the right one as the r value (right value). The l value has to be a variable whereas the l value can be either a constant, a variable, the result of an operation or any combination of these.

Question 7:
What do you mean by statement in Python?
Answer:
A statement is an instruction that the Python interpreter can execute. We have seen two kinds of statements: print and assignment.
When you type a statement on the command line, Python executes it and displays the result, if there is one. The result of a print statement is a value. Assignment statements don’t produce a result.

Question 8:
What do you mean by expression in Python ?
Answer:
An expression is a combination of values, variables, and operators. If you type an expression on the command line, the interpreter evaluates it and displays the result:

Question 9:
Explain Operators and Operands?
Answer:
Operators are special symbols that represent computations like addition and multiplication. The values the operator uses are called Operands.

Question 10:
Use IDLE to calculate : **[CBSE Text Book]**

(a) 6 + 4 * 10
(b) (6 + 4) * 10

Answer:

(a) 6 + 40 = 46
(b) 10 * 10 = 100

Question 11:
What will be the output of the following code : **[CBSE Text Book]**
a = 3 – 4 + 10 b = 5 * 6 c = 7.0/8.0
Answer:
Print “These are the values a, b, c.
These are the values : 9300.0.

Long Answer Type Questions (5 marks each)

Question 1:
Write a example of comparison operators.
Answer:
Following example to understand all the comparison operators available in Python programming language:

```
# !/usr/bin/python
a = 21
b = 10
```

```
c = 0
if (a = = b ):
print “Line 1 – a is equal to b”
else:
print “Line 1 – a is not equal to b”
if ( a ! = b ):
print “Line 2 – a is not equal to b”
else:
print “Line 2 – a is equal to b”
if (a <> b):
print “Line 3 – a is not equal to b”
else:
print “Line 3 – a is equal to b”
if ( a < b ):
print “Line 4 – a is less than b”
else:
print “Line 4 – a is not less than b”
if (a > b ):
print “Line 5 – a is greater than b”
else:
print “Line 5 – a is not greater than b”
a = 5;
b = 20;
if (a < = b):
print “Line 6 – a is either less than or equal to b”
else:
print “Line 6 – a is neither less than nor equal to b”
if (b > = a):
print “Line 7 – b is either greater than or equal to b”
else:
print “Line 7 – b is neither greater than nor equal to b”
```

When you execute the above program it produces following result:

```
Line 1 – a is not equal to b
Line 2 – a is not equal to b
Line 3 – a is not equal to b
Line 4 – a is not less than b
Line 5 – a is greater than b
Line 6 – a is either less than or equal to b
Line 7 – b is either greater than or equal to b
```

Question 2:
Give an example of arithmetic operators.
Answer:
Following example to understand all the arithmetic operators available in Python programming language:

```
# !/usr/bin/python
a = 21
b = 10
c = 0
c = a + b
print “Line 1 – Value of c is “, c
c = a -b
print “Line 2 – Value of c is “, c
c = a * b
print “Line 3 – Value of c is “, c
c = a/b
print “Line 4 – Value of c is “, c
c = a % b
print “Line 5 – Value of c is “, c
a = 2
b = 3
c = a**b
print “Line 6 – Value of c is “, c
a = 10
b = 5
c = a//b
print “Line 7 – Value of c is “, c
```

When you execute the above program it produces following result:

```
Line 1 – Value of c is 31
Line 2 – Value of c is 11
Line 3 – Value of c is 210
Line 4 – Value of c is 2
Line 5 – Value of c is 1
Line 6 – Value of c is 8
Line 7 – Value of c is 2
```

Question 3:
Write a program in Python to explain assignment operators

Answer:
Following example to understand all the assignment operators available in Python programming language :

```
# !/usr/bin/python
a = 21
b = 10
c = 0

c = a + b
print “Line 1 – Value of c is “, c

c += a
print “Line 2 – Value of c is “, c

c *= a
print “Line 3 – Value of c is “, c

c/= a
print “Line 4 – Value of c is “, c

c = 2
c %= a
print “Line 5 – Value of c is “, e

c**= a
print “Line 6 – Value of c is “, c

c//= a
print “Line 7 – Value of c is “, c
```

When you execute the above program it produces following result:

```
Line 1 – Value of c is 31
Line 2 – Value of c is 52
Line 3 – Value of c is 1092
Line 4 – Value of c is 52
Line 5 – Value of c is 2
Line 6 – Value of c is 2097152
Line 7 – Value of c is 99864
```

Question 4:
Write source code in Python to explain bitwise operators.
Answer:
Following example to understand all the bitwise operators available in Python programming language:

```
# !/usr/bin/python

a = 60 # 60 = 0011 1100
b = 13 # 13 = 0000 1101
c = 0

c = a & b; # 12 = 0000 1100
print “Line 1 – Value of c is “, c

c = a | b; # 61 = 0011 1101
print “Line 2 – Value of c is “, c

c = a ^ b; # 49 = 0011 0001
print “Line 3 – Value of c is “, c

c = ~a; # -61 = 1100 0011
print “Line 4 – Value of c is “, c

c = a < < 2; #240 = 1111 0000
print “Line 5 – Value of c is “, c

c = a > > 2; # 15 = 00001111
print “Line 6 – Value of c is “, c
```

When you execute the above program it produces following result:

```
Line 1 – Value of c is 12
Line 2 – Value of c is 61
Line 3 – Value of c is 49
Line 4 – Value of c is -61
Line 5 – Value of c is 240
Line 6 – Value of c is 15
```

Question 5:
Write a program in Python to explain logical operators.
Answer:
Following example to understand all the logical operators available in Python programming language:

```
#!/usr/b in/python

a = 10
b = 20
c = 0
if (a and b):
print “Line 1 – a and b are true”
else:
print “Line 1 – Either a is not true or b is not true”

if (a or b):
print “Line 2 – Either a is true or b is true or both are true”
else:
print “Line 2 – Neither a is true nor b is true”

a = 0
if ( a and b ):
print “Line 3 – a and b are true”
else:
print “Line 3 – Either a is not true or b is not true”

if ( a or b ):
print “Line 4 – Either a is true or b
is true or both are true”
else:
print “Line 4 – Neither a is true nor b is true”
if not( a and b) :
print “Line 5 – a and b are true” else:
print “Line 5 – Either a is not true or b is not true”
```

When you execute the above program it produces following result:

```
Line 1 – a and b are true
Line 2 – Either a is true or b is true or both are true
Line 3 – Either a is not true or b is not true
```

Line 4 – Either a is true or b is true or both are true
Line 5 – a and b are true

Question 6:
Write source in Python code for membership operators.
Answer:
Following example to understand all the membership operators available in Python programming language :

```
#!/usr/bin/python
a = 10
b = 20
list = [1,2,3,4, 5 ];

if ( a in list):
print “Line 1 – a is available in the given list”
else:
print “Line 1 – a is not available in the given list”

if (b not in list):
print “Line 2 – b is not available in the given list”
else:
print “Line 2 – b is available in the given list”

a = 2
if ( a in list):
print “Line 3 – a is available in the given list”
else:
print “Line 3 – a is not available in the given list”
```

When you execute the above program it produces following result:

```
Line 1 – a is not available in the given list
Line 2 – b is not available in the given list
Line 3 – a is available in the given list
```

Question 7:
Write source in Python code for identity operators.
Answer:
Following example to understand all the identity operators available in Python programming language :

```
# !/usr/bin/py thon

a = 20
b = 20

if (a is b):
print “Line 1 – a and b have same
identity”
else:
print “Line 1 – a and b do not have
same identity”
if (id(a) = = id(b)):
print “Line 2 – a and b have
same identity”
else:
print “Line 2 – a and b do not have same identity”
b = 30
if (a is b ):
print “Line 3 – a and b have same
identity”
else:
print “Line 3 – a and b do not have same identity”

if (a is not b ):
print “Line 4 – a and b do not have
same identity”
else:

print “Line 4 – a and b have same identity”
```

When you execute the above program it produces following result:

```
Line 1 – a and b have same identity
Line 2 – a and b have same identity
Line 3 – a and b do not have same identity
Line 4 – a and b do not have same identity
```

Question 11:

Give example to understand operator precedence available in Python programming language.

Answer:
Following example to understand operator precedence available in Python programming language :

```
#!/usr/bin/python

a = 20
b = 10
c = 15
d = 5
e = 0
e = (a + b)*c/d #(30*15)/5
print “Value of (a + b) * c / d is “, e

e = ((a + b) * c)/d #(30*15)/5
print “Value of ((a + b) * c) / d is “, e

e = (a + b) * (c / d); # (30) * (15/5)
print “Value of (a + b) * (c / d) is “, e

e = a + (b*c)/d; # 20 + (150/5)
print “Value of a + (b * c) / d is “, e
```

When you execute the above program it produces following result:

```
Value of (a + b) * c / d is 90
Value of ((a + b) * c) / d is 90
Value of (a + b) * (c / d) is 90
Value of a + (b * c) / d is 50
```

Question 12:
How are comments written in a program ?
Answer:
As the program gets bigger, it becomes difficult to read it, and to make out what it is doing by just looking at it. So it is good to add notes to the code, while writing it. These notes are known as comments. In Python, comment start with ‘#’ symbol. Anything written after # in a line is ignored by interpreter, i.e. it will not have any effect on the program.
For example,
calculating area of a square > > > area = side * * 2
For adding multi-line comment in a program :
(i) Place ‘#’ in front of each line, or,

(ii) Use triple quoted string. They will only work as comment, when they are not being used as docstring.