ASSIGNMENT

Q: Define 68 built-in functions of python along with programs for implementating them and screenshots of output.You may combine few functions in same program.

1. abs()

Definition: Returns the absolute value of a number.

```
print(abs(-10))
```

Output:

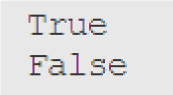
```
10
```

2. all()

Definition: Returns True if all elements in an iterable are True.

```
print(all([True, 1, 'Hello']))
print(all([True, 0, '']))
```

Output:



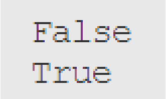
```
True
False
```

3. any()

Definition: Returns True if at least one element in an iterable is True.

```
print(any([False, 0, ""]))  # Output: False
print(any([False, 1, ""]))  # Output: True
```

Output:



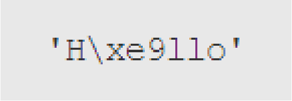
```
False
True
```

4. ascii()

Definition: Returns a readable string representation of an object, escaping non-ASCII characters.

```
print(ascii('Héllo'))  # Output: 'H\xe9llo'
```

Output:



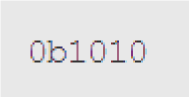
```
'H\xe9llo'
```

5. bin()

Definition: Converts an integer to its binary representation.

```
print(bin(10))  # Output: '0b1010'
```

Output:



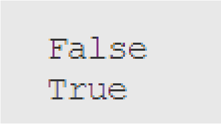
```
0b1010
```

6. bool()

Definition: Converts a value to a Boolean.

```
print(bool(0))  # Output: False
print(bool(1))  # Output: True
```

Output:



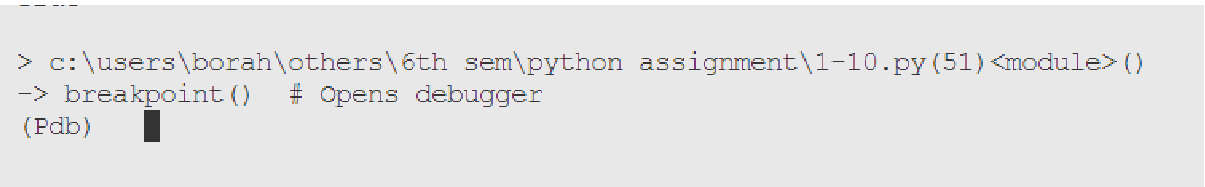
```
False
True
```

7. breakpoint()

Definition: Enters Python’s interactive debugger.

```
x = 10
breakpoint()  # Opens debugger
print(x)
```

Output:



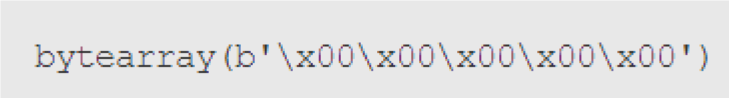
```
> c:\users\borah\others\6th sem\python assignment\1-10.py(51)<module>()
-> breakpoint()  # Opens debugger
(Pdb)
```

8. bytearray()

Definition: Returns a mutable sequence of bytes.

```
b = bytearray(5)
print(b)  # Output: bytearray(b'\x00\x00\x00\x00\x00')
```

Output:



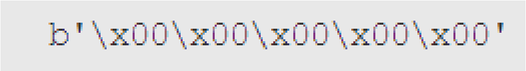
```
bytearray(b'\x00\x00\x00\x00\x00')
```

9. bytes()

Definition: Returns an immutable bytes object.

```
b = bytes(5)
print(b)  # Output: b'\x00\x00\x00\x00\x00'
```

Output:



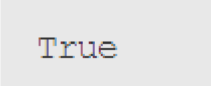
```
b'\x00\x00\x00\x00\x00'
```

10. callable()

Definition: Checks if an object is callable.

```
def func():
    pass

print(callable(func))  # Output: True
```

Output:



```
True
```

11. chr()

Definition: Converts an integer to a character.

```
print(chr(65))  # Output: 'A'
```

Output:

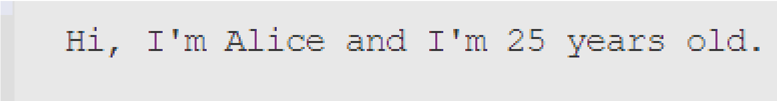
```
A
```

12. __init__()

Definition: The __init__() method is a special constructor method in Python classes. It is automatically called when a new object of a class is created, and it is used to initialize attributes of the object.

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def introduce(self):
        print(f"Hi, I'm {self.name} and I'm {self.age} years old.")

# Creating an object of the Person class
person1 = Person("Alice", 25)
person1.introduce()
```

Output:



```
Hi, I'm Alice and I'm 25 years old.
```

13. compile()

Definition: Compiles a string into a code object.

```
code = compile('print("Hello")', '', 'exec')
exec(code)  # Output: Hello
```

Output:

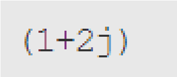
```
Hello
```

14. complex()

Definition: Returns a complex number.

```
print(complex(1, 2))  # Output: (1+2j)
```

Output:



```
(1+2j)
```

15. delattr()

Definition: Deletes an attribute from an object.

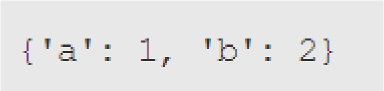
```
class A:
    x = 10

delattr(A, 'x')
```

16. dict()

Definition: Creates a dictionary.

```
d = dict(a=1, b=2)
print(d)  # Output: {'a': 1, 'b': 2}
```

Output:



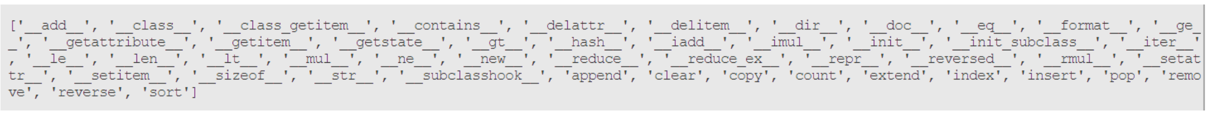
```
{'a': 1, 'b': 2}
```

17. dir()

Definition: Returns a list of attributes and methods of an object.

```
print(dir([]))  # Output: List of methods available for lists
```

Output:



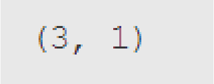
```
['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge_
_', '__getattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__'
, '__le__', '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setat
tr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remo
ve', 'reverse', 'sort']
```

18. divmod()

Definition: Returns quotient and remainder of division.

```
print(divmod(10, 3))  # Output: (3, 1)
```

Output:



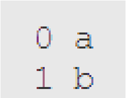
```
(3, 1)
```

19. enumerate()

Definition: Returns an enumerate object.

```
for i, v in enumerate(['a', 'b']):
    print(i, v)
```

Output:



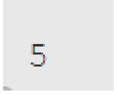
```
0 a
1 b
```

20. eval()

Definition: Evaluates a string as Python code.

```
print(eval('2 + 3'))  # Output: 5
```

Output:



```
5
```

21. exec()

Definition: Executes a block of code.

```
exec('print("Hello, World!")')
```

Output:

```
● Hello, World!
```

22. filter()

Definition: Filters elements of an iterable.

```
nums = [1, 2, 3, 4]
print(list(filter(lambda x: x % 2 == 0, nums)))  # Output: [2, 4]
```

Output:

```
[2, 4]
```

23. float()

Definition: Converts a value to a float.

```
print(float(10))  # Output: 10.0
```

Output:

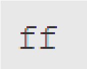
```
10.0
```

24. format()

Definition: Formats a value.

```
print(format(255, 'x'))  # Output: 'ff'
```

Output:



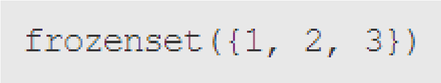
```
ff
```

25. frozenset()

Definition: Returns an immutable set.

```
fs = frozenset([1, 2, 3])
print(fs)  # Output: frozenset({1, 2, 3})
```

Output:



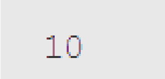
```
frozenset({1, 2, 3})
```

26. getattr()

Definition: Gets an attribute from an object.

```
class Test:
    x = 10

obj = Test()
print(getattr(obj, 'x'))  # Output: 10
```

Output:



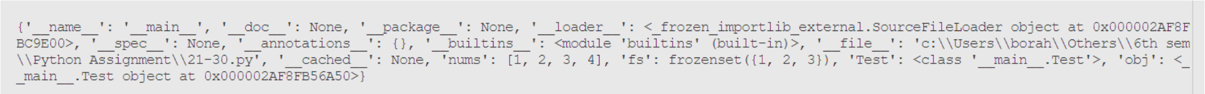
```
10
```

27. globals()

Definition: Returns a dictionary of global variables.

```
print(globals())
```

Output:



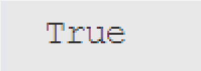
```
{'__name__': '__main__', '__doc__': None, '__package__': None, '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x000002AF8F
BC9E00>, '__spec__': None, '__annotations__': {}, '__builtins__': <module 'builtins' (built-in)>, '__file__': 'c:\\Users\\borah\\Others\\6th sem
\\Python Assignment\\21-30.py', '__cached__': None, 'nums': [1, 2, 3, 4], 'fs': frozenset({1, 2, 3}), 'Test': <class '__main__.Test'>, 'obj': <_
_main__.Test object at 0x000002AF8FB56A50>}
```

28. hasattr()

Definition: Checks if an object has an attribute.

```
print(hasattr(Test, 'x'))  # Output: True
```

Output:



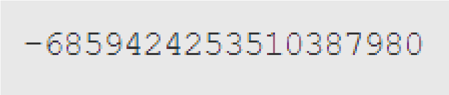
```
True
```

29. hash()

Definition: Returns the hash of an object.

```
print(hash("hello"))
```

Output:



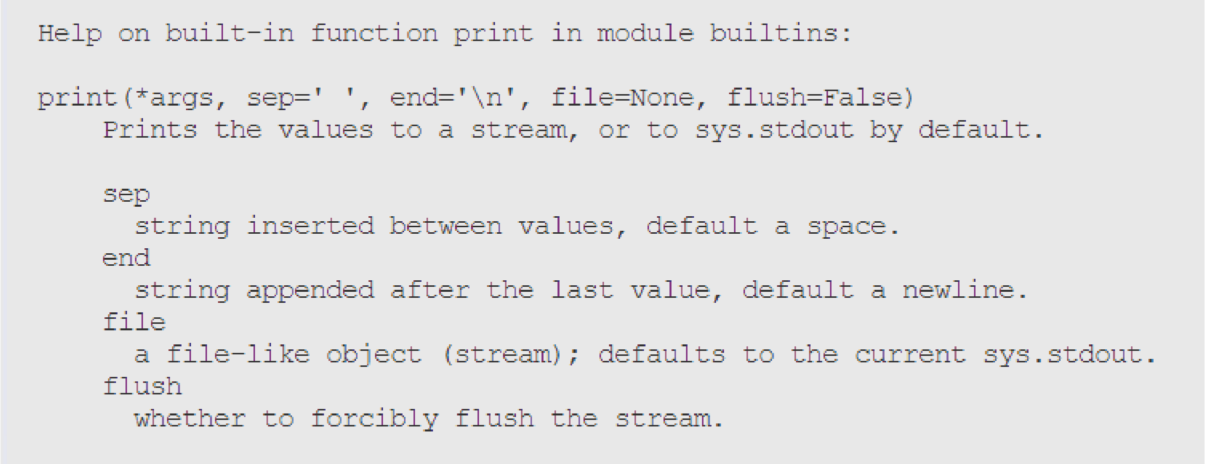
```
-6859424253510387980
```

30. help()

Definition: Displays help information.

```
help(print)
```

Output:



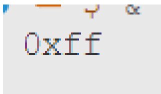
```
Help on built-in function print in module builtins:

print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by default.

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newline.
    file
      a file-like object (stream); defaults to the current sys.stdout.
    flush
      whether to forcibly flush the stream.
```

31. hex()

Definition: Converts an integer to a hexadecimal string.

```
print(hex(255))  # Output: '0xff'
```

Output:



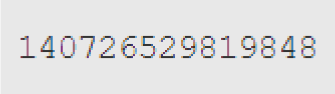
```
0xff
```

32. id()

Definition: Returns the unique identifier (memory address) of an object.

```
x = 10
print(id(x))  # Output: A unique integer (memory address)
```

Output:



```
140726529819848
```

33. __import__()

Definition: Dynamically imports a module by name (not commonly used directly).

```
math_module = __import__('math')
print(math_module.sqrt(16))  # Output: 4.0
```

Output:

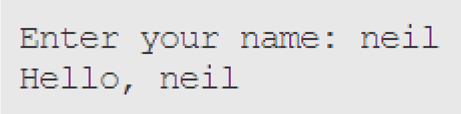
```
4.0
```

34. input()

Definition: Reads a line of input from the user.

```
name = input("Enter your name: ")
print("Hello,", name)
```

Output:



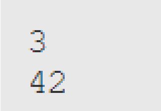
```
Enter your name: neil
Hello, neil
```

35. int()

Definition: Converts a value to an integer.

```
print(int(3.7))  # Output: 3
print(int("42"))  # Output: 42
```

Output:



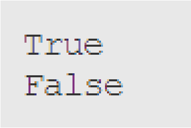
```
3
42
```

36. isinstance()

Definition: Checks if an object is an instance of a class.

```
print(isinstance(10, int))  # Output: True
print(isinstance("hello", int))  # Output: False
```

Output:



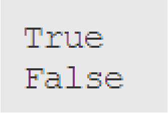
```
True
False
```

37. issubclass()

Definition: Checks if a class is a subclass of another.

```
class A: pass
class B(A): pass

print(issubclass(B, A))  # Output: True
print(issubclass(A, B))  # Output: False
```

Output:



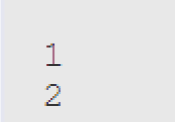
```
True
False
```

38. iter()

Definition: Returns an iterator object.

```
lst = [1, 2, 3]
it = iter(lst)
print(next(it))  # Output: 1
print(next(it))  # Output: 2
```

Output:



```
1
2
```

39. len()

Definition: Returns the length of an object.

```
print(len([1, 2, 3]))  # Output: 3
print(len("hello"))  # Output: 5
```

Output:

```
3
5
```

40. list()

Definition: Creates a list.

```
print(list("hello"))  # Output: ['h', 'e', 'l', 'l', 'o']
```

Output:

```
['h', 'e', 'l', 'l', 'o']
```

41. locals()

Definition: Returns a dictionary of local variables.

```
def func():
    x = 10
    print(locals())

func()  # Output: {'x': 10}
```

Output:

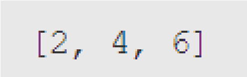
```
{'x': 10}
```

42. map()

Definition: Applies a function to all items in an iterable.

```
nums = [1, 2, 3]
print(list(map(lambda x: x * 2, nums)))  # Output: [2, 4, 6]
```

Output:



```
[2, 4, 6]
```

43. max()

Definition: Returns the largest item in an iterable.

```
print(max([1, 2, 3]))  # Output: 3
```

Output:

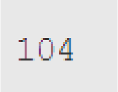
```
3
```

44. memoryview()

Definition: Returns a memory view object.

```
b = bytearray("hello", "utf-8")
m = memoryview(b)
print(m[0])  # Output: 104 (ASCII of 'h')
```

Output:



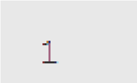
```
104
```

45. min()

Definition: Returns the smallest item in an iterable.

```
print(min([1, 2, 3]))  # Output: 1
```

Output:



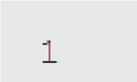
```
1
```

46. next()

Definition: Retrieves the next item from an iterator.

```
it = iter([1, 2, 3])
print(next(it))  # Output: 1
```

Output:



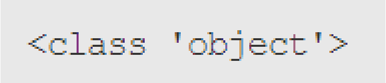
```
1
```

47. object()

Definition: Returns a new object.

```
obj = object()
print(type(obj))  # Output: <class 'object'>
```

Output:



```
<class 'object'>
```

48. oct()

Definition: Converts an integer to an octal string.

```
print(oct(10))  # Output: '0o12'
```

Output:

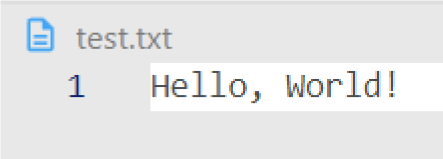
```
0o12
```

49. open()

Definition: Opens a file.

```
with open("test.txt", "w") as f:
    f.write("Hello, World!")
```

Output:



test.txt

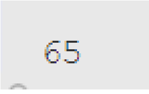
```
1    Hello, World!
```

50. ord()

Definition: Returns the Unicode code point of a character.

```
print(ord('A'))  # Output: 65
```

Output:



```
65
```

51. pow()

Definition: Returns the power of a number.

```
print(pow(2, 3))  # Output: 8
print(pow(2, 3, 5))  # Output: 3 (2^3 % 5)
```

Output:

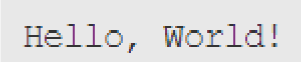
```
8
3
```

52. print()

Definition: Prints output to the console.

```
print("Hello, World!")
```

Output:



```
Hello, World!
```

53. property()

Definition: Returns a property attribute.

```
class Test:
    def __init__(self, x):
        self._x = x

    @property
    def x(self):
        return self._x

obj = Test(10)
print(obj.x)  # Output: 10
```

Output:

```
10
```

54. range()

Definition: Generates a sequence of numbers.

```
print(list(range(1, 5)))  # Output: [1, 2, 3, 4]
```

Output:

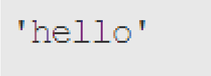
```
[1, 2, 3, 4]
```

55. repr()

Definition: Returns a string representation of an object.

```
print(repr("hello"))  # Output: "'hello'"
```

Output:



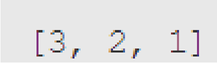
```
'hello'
```

56. reversed()

Definition: Returns a reversed iterator.

```
print(list(reversed([1, 2, 3])))  # Output: [3, 2, 1]
```

Output:



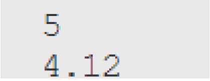
```
[3, 2, 1]
```

57. round()

Definition: Rounds a number to the nearest integer.

```
print(round(4.6))  # Output: 5
print(round(4.123, 2))  # Output: 4.12
```

Output:



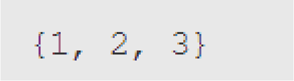
```
5
4.12
```

58. set()

Definition: Creates a set.

```
print(set([1, 2, 3, 3]))  # Output: {1, 2, 3}
```

Output:



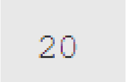
```
{1, 2, 3}
```

59. setattr()

Definition: Sets an attribute on an object.

```
class Test:
    x = 10

obj = Test()
setattr(obj, 'x', 20)
print(obj.x)  # Output: 20
```

Output:



```
20
```

60. slice()

Definition: Returns a slice object.

```
lst = [1, 2, 3, 4, 5]
print(lst[slice(1, 4)])  # Output: [2, 3, 4]
```

Output:

```
[2, 3, 4]
```

61. sorted()

Definition: Returns a sorted list.

```
print(sorted([3, 1, 2]))  # Output: [1, 2, 3]
```

Output:

```
[1, 2, 3]
```

62. str()

Definition: Converts an object to a string.

```
print(str(100))  # Output: '100' print(str(100))  # Output: '100'
```

Output:

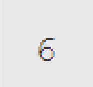
```
100
```

63. sum()

Definition: Returns the sum of an iterable.

```
print(sum([1, 2, 3]))  # Output: 6
```

Output:



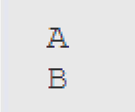
```
6
```

64. super()

Definition: Calls a method from the parent class.

```
class A:
    def show(self):
        print("A")

class B(A):
    def show(self):
        super().show()
        print("B")

obj = B()
obj.show()
```

Output:



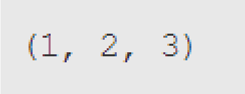
```
A
B
```

65. tuple()

Definition: Creates a tuple.

```
print(tuple([1, 2, 3]))  # Output: (1, 2, 3)
```

Output:



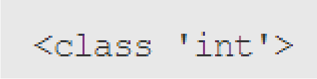
```
(1, 2, 3)
```

66. type()

Definition: Returns the type of an object.

```
print(type(10))  # Output: <class 'int'>
```

Output:



```
<class 'int'>
```

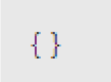
67. vars()

Definition: Returns the __dict__ attribute of an object.

```
class Test:
    x = 10
```

```
obj = Test()
print(vars(obj))
```

Output:



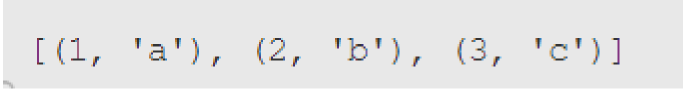
```
{}
```

68. zip()

Definition: Combines multiple iterables.

```
a = [1, 2, 3]
b = ['a', 'b', 'c']
print(list(zip(a, b)))  # Output: [(1, 'a'), (2, 'b'), (3, 'c')]
```

Output:



```
[(1, 'a'), (2, 'b'), (3, 'c')]
```