1. Bug in a program that causes it to operate incorrectly, but not to terminate abnormally.

2. Occurs during the execution of a program, that disrupts the normal flow of the program's instructions

3. Try - Finally clause
   except clause with no exceptions
   The except clause with multiple exceptions
   Try - except - else

4.
```
try:
    - - - -
    - - - -
except Exception I:
    - - - - - -
except Exception II:
    - - - - - -
else:
    - - - -
```

5. A module allows you to logically organize your python code. Grouping related code into a module makes the codes easier to understand and use.

6. A package is a hierarchial file directory structure that defines a single python application environment that consists of modules and sub packages and sub-sub packages.

7. It is also possible to import all names from a module into a current namespace.

8. block of code which handles the problem as elegantly as possible

10. provides a way of using operating system dependent functionality.
allows you to interface with the underlying operating system that python is running on

11. used for generating random data

12. provides access to the mathematical functions

13. Raised when the builtin function for a datatype has the valid type of arguments, but the arguments have invalid values specified

14. Raised when division or module by zero takes place for all numeric type.

15. Raised when an identifier is not found in the local or global namespace.

16. Error indicates a problem that a resonable application should not try to catch.
Exception indicates condutions that a reasonable application might want to catch.

17. import module1 [, module2 [, ---- moduleN]]

18. code that doesnot need the try-block's protection

19. dictionary of variable names keys and their corresponding objects values

20. rb, r+, rb+, r

21. wb, w+, wb+, w
ab, a+, ab+, a.

22. computer resource for recording data discretely on a computer storage device

23. open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)

24. f = open("test.doc", "r")

25. close the file. A closed file cannot be read or written anymore. Any operation which requires that the file be open will raise a ValueError after the file has been closed.

26. file pointer points at the beginning of the file

27. seek() mathod

28. tell()

29. The seek(offset[, from])

30. The tell()

31. readline()

31. opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists.

32. The open() function opens the file (if possible) and returns a corresponding file object.

33. Read until EOF using readline() and return a list containing the lines thus read. If the optional sizehint argument is present, instead of reading up to EOF, whole lines totalling approx sizehint bytes are read.

34. by using rename() manod

35. using mkdir(path)

39. There are two write() method on file object

write(str);

write a string to the file. There is no return value. the string may not actually show up in the file until the flush() or close() method is called.

writelines(sequence):

write a sequence of strings to the file

40 The seek(offset[, from])

changes the current file position.

41. class variable: A variable that is shared by all instances of a class that characterize any object of the class.
Instance variable: A variable that is defined inside a method and belongs only to the current instance of a class.

42. A special kind of function that is defined in a class definition

43. class ClassName;
'optional class documentation string'
class_suite

44. __dict__
__doc__
__name__
__module__
__bases__

45.

46. __del__() destructor prints the class name of insistance that is about to be destroyed

47. used to automatically to free memory space. The process by which python periodically reclaims blocks of memory that no longer are in use in termed garbage collection.

48. Inheritance, a class is inherited by another class.

49. A particular object belonging to a particular class can be used in the same way as it if were a different object belonging to a different class.

50. initialization method that python calls when you create a new instance of this class.

51. that is invoked when the instance is about to be destroyed.

52. using __del__(self). we can delete an object

53. class DerivedClass(BaseClass):

54. restrict access to methods & variables. This can prevent the data from being modified by accident and is known as encapsulation.

55. subclass to provide a specific implementation of method that is already provided by one of its parent class.

56.

57. attributes with double underscore prefix.

58. refers newly created object

59.

60.

61. using super() function

62.