NAME: HASSAN KAMAL

ID#: 12925

SEMESTER: 5

QUESTION#1:

What type of errors do occur in Python, write the program with different types of errors as well as write separate correction code?

ANSWER:

We can make certain mistakes while writing a program that lead to errors when we try to run it. A python program terminates as soon as it encounters an unhandled error. These errors can be broadly classified into two classes:

1. Syntax errors
2. Logical errors (Exceptions)

Python Syntax Errors

Error caused by not following the proper structure (syntax) of the language is called **syntax error** or **parsing error**.

Let's look at one example:

```
>>> if a < 3
  File "<interactive input>", line 1
    if a < 3
         ^
Syntax Error: invalid syntax
```

As shown in the example, an arrow indicates where the parser ran into the syntax error.

We can notice here that a colon: is missing in the if statement.

Python Logical Errors (Exceptions)

Errors that occur at runtime (after passing the syntax test) are called **exceptions** or **logical errors**.

```
>>> 10 * (1/0)
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
ZeroDivisionError: division by zero
>>> 4 + spam*3
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
NameError: name 'spam' is not defined
>>> '2' + 2
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
TypeError: Can't convert 'int' object to str implicitly
```

```
import sys

try:
   f = open('myfile.txt')
   s = f.readline()
   i = int(s.strip())
except IOError as err:
   print("I/O error: {0}".format(err))
except ValueError:
   print("Could not convert data to an integer.")
except:
   print("Unexpected error:", sys.exc_info()[0])
   raise
```

QUESTION#2:
What are Boolean String test, write the code for each Boolean string test code?

ANSWER:

Boolean Strings

A string in Python can be tested for truth value.

The return type will be in Boolean value (True or False)

```
my_string = "Hello World"

my_string.isalnum()          #check if all char are numbers
my_string.isalpha()          #check if all char in the string are alphabetic
my_string.isdigit()          #test if string contains digits
my_string.istitle()          #test if string contains title words
my_string.isupper()          #test if string contains upper case
my_string.islower()          #test if string contains lower case
my_string.isspace()          #test if string contains spaces
my_string.endswith('d')      #test if string endswith a d
my_string.startswith('H')    #test if string startswith H
```

To see what the return value (True or False) will be, simply print it out.

```
my_string="Hello World"

print my_string.isalnum()              #False
print my_string.isalpha()              #False
print my_string.isdigit()      #False
print my_string.istitle()      #True
print my_string.isupper()              #False
print my_string.islower()              #False
print my_string.isspace()              #False
print my_string.endswith('d')          #True
print my_string.startswith('H')        #True
```

QUESTION#3:
What is formatting string input mean in Python, write a program in which formatting string input is used?

ANSWER:

Python uses C-style string formatting to create new, formatted strings. The "%" operator is used to format a set of variables enclosed in a "tuple" (a fixed size list), together with a format string, which contains normal text together with "argument specifiers", special symbols like "%s" and "%d".

```
>>> '%x' % errno
'badc0ffee'

>>> 'Hey %s, there is a 0x%x error!' % (name, errno)
'Hey Bob, there is a 0xbadc0ffee error!'

>>> 'Hey %(name)s, there is a 0x%(errno)x error!' % {
...     "name": name, "errno": errno }
'Hey Bob, there is a 0xbadc0ffee error!'
```