MIDTERM

Submitted By:

Hamza Khan Afridi (13071)

BS (SE)

Subject :

Data Sciences

Question1 :

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

ANS: In python there are three types of errors; syntax errors, logic errors and exceptions/Runtime errors.

1. Syntax errors
2. Runtime errors
3. Logical errors

1) Syntax errors: Python will find these kinds of errors when it tries to parse your program, and exit with an error message without running anything.

Wrong code:

```
fro i in range(10):
 File "<stdin>", line 1
  fro i in range(10):
```

Correct code:

```
for i in range(10):
 File "<stdin>", line 1
  for I in range(10):
```

2) **Runtime errors:** The second type of error is a runtime error. A program with a runtime error is one that passed the interpreter's syntax checks, and started to execute. However, during the execution of one of the statements in the program, an error occurred that caused the interpreter to stop executing the program and display an error message.

Wrong code:

```
import math
```

```
def square(n):
if n<0:
    return 0
else:
    tst=int(math.sqrt(n))
    if tst*tst==n or  (tst+1)*(tst+1)==n or (tst-1)*(tst-1)==n:
        return 1
t=input()
for i in t:
 n=input()
   flag=square(n)
   if flag==1:
      print "Case %l: Yes\n"%i
    else:
      print "Case %l: No\n"%i
   t=t-1
```

Correct code:

```
Import math
def square(n):
if n<0:
    return 0
else:
    tst=int(math.sqrt(n))
    if tst*tst==n or  (tst+1)*(tst+1)==n or (tst-1)*(tst-1)==n:
        return 1
```

```
t=input()
for i in range(t):
 n=input()
   flag=square(n)
   if flag==1:
       print "Case %l: Yes\n"%i
    else:
       print "Case %l: No\n"%i
    t=t-1
```

3) **Logical errors:** Logical errors are the most difficult to fix. They occur when the program runs without crashing, but produces an incorrect result. The error is caused by a mistake in the program's logic. You won't get an error message, because no syntax or runtime error has occurred. You will have to find the problem on your own.

Wrong code:

```
x = float(input('Enter a number: '))
y = float(input('Enter a number: '))
z = x+y/2
print ('The average of the two numbers you have entered is:',z)
```

Correct code:

```
x = float(input('Enter a number: '))
y = float(input('Enter a number: '))
z =( x+y)/2
print ('The average of the two numbers you have entered is:',z)
```

Question2 :

What are Boolean String test, write the code for each Boolean string test code?

ANS: Boolean values are the two constant objects False and True. They are used to represent truth values (other values can also be considered false or true). In numeric contexts (for example, when used as the argument to an arithmetic operator), they behave like the integers 0 and 1, respectively. The built-in function bool() can be used to cast any value to a Boolean, if the value can be interpreted as a truth value.They are written as False and True, respectively.A string in Python can be tested for truth value.The return type will be in Boolean value (True or False).Let’s make an example, by first create a new variable and give it a value.

Code:

```
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
```

Question3 :

What is formatting string input mean in Python, write a program in which formatting string input is used?

ANS: 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".

Let's say you have a variable called "name" with your user name in it, and you would then like to print(out a greeting to that user.):

```
# This prints out "Hello, John!"

name = "John"

print("Hello, %s!" % name)
```

To use two or more argument specifiers, use a tuple (parentheses):

```
# This prints out "John is 23 years old."
```

```
name = "John"
age = 23
print("%s is %d years old." % (name, age))
```

THE END