Python

Python is just the language for you. Python is simple to use, but it is a real programming language, offering much more structure and support for large programs than shell scripts or batch files can offer. On the other hand, Python also offers much more error checking than C, and, being a *very-high-level language*, it has high-level data types built in, such as flexible arrays and dictionaries. Because of its more general data types Python is applicable to a much larger problem domain than Awk or even Perl, yet many things are at least as easy in Python as in those languages. Python enables programs to be written compactly and readably. Programs written in Python are typically much shorter than equivalent C, C++, or Java programs, for several reasons:

- the high-level data types allow you to express complex operations in a single statement;
- statement grouping is done by indentation instead of beginning and ending brackets;
- no variable or argument declarations are necessary.

An Informal Introduction to Python:

Using Python as a Calculator

Let's try some simple Python commands. Start the interpreter and wait for the primary prompt, `>>>`. (It shouldn't take long.)

Numbers

The interpreter acts as a simple calculator: you can type an expression at it and it will write the value. Expression syntax is straightforward: the operators `+`, `-`, `*` and `/` work just like in most other languages (for example, Pascal or C); parentheses (`()`) can be used for grouping. For example:

```
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5*6) / 4
5.0
>>> 8 / 5  # division always returns a floating point number
1.6

# Simple printing as calculator
>>> x=10
>>> x
10
>>>
```

The integer numbers (e.g. `2`, `4`, `20`) have type `int`, the ones with a fractional part (e.g. `5.0`, `1.6`) have type `float`. We will see more about numeric types later in the tutorial.

Division (`/`) always returns a float. To do floor division and get an integer result (discarding any fractional result) you can use the `//` operator; to calculate the remainder you can use `%;`

`Note: '#' is used for Comment`:

```
>>> 17 / 3  # classic division returns a float
5.666666666666667
>>>
>>> 17 // 3  # floor division discards the fractional part
5
>>> 17 % 3  # the % operator returns the remainder of the division
2
>>> 5 * 3 + 2  # result * divisor + remainder
17
```

With Python, it is possible to use the `**` operator to calculate powers:

```
>>> 5 ** 2  # 5 squared
25
>>> 2 ** 7  # 2 to the power of 7
128
```

The equal sign (`=`) is used to assign a value to a variable. Afterwards, no result is displayed before the next interactive prompt:

```
>>> width = 20
>>> height = 5 * 9
>>> width * height
900
```

If a variable is not “defined” (assigned a value), trying to use it will give you an error:

```
>>> n  # try to access an undefined variable
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
Name Error: name 'n' is not defined
```

There is full support for floating point; operators with mixed type operands convert the integer operand to floating point:

```
>>> 4 * 3.75 - 1
14.0
```

In interactive mode, the last printed expression is assigned to the variable `_`. This means that when you are using Python as a desk calculator, it is somewhat easier to continue calculations, for example:

```
>>> tax = 12.5 / 100
>>> price = 100.50
>>> price * tax
12.5625
>>> price + _
113.0625
>>> round(_, 2)
113.06
```

This variable should be treated as read-only by the user. Don't explicitly assign a value to it — you would create an independent local variable with the same name masking the built-in variable with its magic behavior.

In addition to `int` and `float`, Python supports other types of numbers, such as `Decimal` and `Fraction`. Python also has built-in support for complex numbers, and uses the `j` or `J` suffix to indicate the imaginary part (e.g. `3+5j`).

Input and Output

There are several ways to present the output of a program; data can be printed in a human-readable form, or written to a file for future use. This chapter will discuss some of the possibilities.

Output by **print** statement

```
print('This sentence is output to the screen')
# Output:
This sentence is output to the screen
a = 5
print('The value of a is', a)
# Output:
The value of a is 5
```

Input by **input** statement

```
>>> num = input('Enter a number: ')
Enter a number: 10
>>> num
'10
#Prints the value of num
```

Programming Fundamentals

Of course, we can use Python for more complicated tasks than adding two and two together. For instance, we can write an initial sub-sequence of the Fibonacci series as follows:

#1. WAP to find the sum of any 2 numbers

```
a=int(input("Enter the value of a "))
b=int(input("Enter the value of b "))
s=a+b
print ("Sum is =>",s)
```

#2. WAP to find the sum of any 2 numbers

```
a=int(input("Enter the value of a "))
b=int(input("Enter the value of b "))
s=a+b
print ("Sum is =>",s)
```

#3. WAP to find the division of any 2 numbers

```
a=int(input("Enter the value of a "))
b=int(input("Enter the value of b "))
s=a/b
print ("Quotient is =>",s)
```

#4. WAP to find the product of any 2 numbers

```
a=int(input("Enter the value of a "))
b=int(input("Enter the value of b "))
s=a*b
print ("Product is =>",s)
```

#5. WAP to find the Area and Perimeter of Rectangle

```
x=int(input("Enter the value of Length "))
y=int(input("Enter the value of Breadth "))
a=x*y
r=2*(x+y)
print ("Area is =>",a)
print ("Perimeter is =>",r)
```

#6 WAP to find the greater number of any 2 numbers

```
N1=input("F No")
N2=input("L No")
N3=input("3rd no")
if (N1>N2):
    g=N1
else:
    g=N2
s=(int)(N1+N2)
print ("sum",s)
print("Greater no",g)
```

#7 WAP to find the greatest number among 3 numbers

```
a=int(input("Enter the value of a"))
b=int(input("Enter the value of b"))
c=int(input("Enter the value of c"))
if(a==b) and (b==c):
        print("all are equal")
elif(a>b) and (a>c):
        g=a
elif(b>a) and (b>c):
        g=b
else:
        g=c

        print ("Greatest is =>",g)
```

#8 WAP to check if the number is positive, print an appropriate message

```
num = 3
if num > 0:
    print(num, "is a positive number.")
print("This is always printed.")

num = -1
if num > 0:
    print(num, "is a positive number.")
print("This is also always printed.")
```

#9 WAP to check if the number is positive or negative

```
num = 3
if num >= 0:
    print("Positive or Zero")
else:
    print("Negative number")
```

#10 WAP to check if the number is positive or negative using multiple if….else

```
num = float(input("Enter a number: "))
if num >= 0:
    if num == 0:
        print("Zero")
    else:
        print("Positive number")
else:
    print("Negative number")
```

#11 WAP to check if the number is positive or negative using if….elif…else
```
num = float(input("Enter a number: "))
if num > 0:
    print("Positive number")
elif num == 0:
    print("Zero")
else:
    print("Negative number")
```

#12 Python program to check if the input number is odd or even.
```
num = int(input("Enter a number: "))
if (num % 2) == 0:
   print(num , "is Even")
else:
   print(num ,"is Odd")
```

#13 Python program to check if the input year is a leap year or not
```
year = int(input("Enter a year: "))
if (year % 4) == 0:
   if (year % 100) == 0:
       if (year % 400) == 0:
           print(year ,"is a leap year")
       else:
           print(year,"is not a leap year")
   else:
       print(year,"is a leap year")
else:
   print("{0} is not a leap year".format(year))
```

#14 Program to check the character is vowel
```
x=input('Enter a character ')
if (x=='a' or x=='A'):
        print (x,'is vowel')
if(x=='e' or x=='E'):
        print(x,'is vowel')
if(x=='i' or x=='I'):
        print(x,'is vowel')
if(x=='o' or x=='O'):
        print(x,'is vowel')
if(x=='u' or x=='U'):
        print(x,'is vowel')
```

LOOPING

#15 WAP to find sum of n

```
n = int(input("Enter n: "))
sum = 0
i = 1
while i <= n:
        sum = sum + i
        i = i+1    # increment

print("The sum is", sum)
```

Output:
```
Enter n: 4
The sum is 10
```

#16 WAP to find sum of n

```
n = int(input("Enter n: "))
sum = 0
i = 1

while i <= n:
        sum = sum + i
        i = i+1    # increment
        print("The value is",i)
```

Output:
```
Enter n: 4
The value is 2
The value is 3
The value is 4
The value is 5
```

#17 WAP to find the sum of all numbers
```
n = int(input("Enter n: "))
sum = 0
i = 1

while i <= n:
   sum = sum + i
   i = i+1    # increment
print("The sum is", sum)
```

Output:
```
Enter n: 5
The sum is 1    2
The sum is 3    3
The sum is 6    4
The sum is 10    5
The sum is 15    6
The sum is 15    6
```

#18 WAP to display:
```
***
***
***
```
```
n = int(input("Enter n: "))
i=1
while i<=n:
        x="*" *n
        print(x)
        i=i+1
```

#19 WAP to display:

```
 *
**
***
```

```
n = int(input("Enter n: "))
i=1
while i<=n:
    x="*" * i
    print(x)
    i=i+1
```

#20 WAP to display:

```
  *
 **
***
```

```
n = int(input("Enter n: "))
i=1
j=n-1
while i<n:
    y=" "*j
    x="*" * i
    print(y,x)
    i=i+1
    j=j-1
```