Built-in subprograms

Example	Description
character = chr (65) print (character)	This example returns the string which matches the Unicode value of 65. The output is: A
firstNumber = 7 secondNumber = 21 print(bool(firstNumber == secondNumber))	This example returns False as firstNumber is not equal to secondNumber.
name = "Manjit" print(bool(name))	This example returns True as name is a non-empty string.
newList = [] print(bool(newList))	This example returns False as newList is an empty list.
decimalNumber = float (input (print (decimalNumber)	This example converts the string input decimalNumber into a real.
name = input ("Name: ") print (name)	This example displays the prompt to the screen and waits for the user to type in characters followed by a new line. The console session looks like this: Name: *Shaun* Shaun
integerNumber = int (input(print (integerNumber)	This example converts the string input into an integer.
word = "Garage" length = len (word) print (length)	This example returns the length of the word 'Garage'. The output is: 6
string = ord ("K") print (string)	This example returns the integer equivalent to the Unicode single character string 'K'. The output is: 75
print ("Hello")	This example prints 'Hello' to the display. The output is: Hello
for num in range (2): print (num)	This example prints the numbers from 0 up to, but not including 2, incrementing 1 each time. The output is: 0 1
for num in range (10, 13): print (num)	This example prints the numbers from 10 up to, but not including 13, incrementing 1 each time. The output is: 10 11 12
for num in range (10, 0, -2): print (num)	This example prints the numbers from 10 down to, but not including 0, decreasing by 2 each time. The output is: 10 8 6 4 2

Example	Description
total = 62.7259 cost = round (total, 2) print (cost)	This example rounds the variable 'total' to two decimal places and stores the result in the variable 'cost'. The output is: 62.73
number = 66 stringNumber = str (number) print (stringNumber + " is stored as a string.")	Python can only concatenate string objects. So, if the programmer wants to merge a string with a nonstring (an integer or a float), they must use the function 'str()' to convert the non-string to a string.

List subprograms

This table shows examples of using the list subprograms.

Example	Description
table = [1, 2, 3, 4] table.append (5) print (table)	This example appends an item to an existing list. The output is: [1, 2, 3, 4, 5]
table = [10, 9, 8, 7, 6] del table[1] print (table)	This example deletes the item at index 1. The output is: [10, 8, 7, 6]
table = [1, 2, 4, 5] table.insert (2, 3) print (table)	This example inserts a new item, the number 3, at index position 2. The output is: [1, 2, 3, 4, 5]
table_one = list () print (table_one) table_two = [] print (table_two)	This example shows two ways to create an empty list. Once created the append subprogram can be used to put items into the list. The output is: [] []

Library modules

The PLS specifies a limited set of library modules and an even smaller set of actual subprograms found in them that students are expected to be familiar with.

This table shows how to import a library module.

Statement	Example	Description
import	import turtle	This example imports the turtle graphics library module.
import	from math import pi	This example imports the constant Pi from the math library.

Random library module

This table shows examples of using the random library module subprograms.

Example	Description
import random num = random.randint (1, 10) print (num)	This example shows how to generate a random whole number between two bounds, inclusive. The output is: 8
import random decimal = random.random () print (decimal)	This example shows how to generate a random decimal number, between 0 and 1. The output is: 0.9754469153477409

Math library module

This table shows examples of using the math library module subprograms and constant.

Example	Description
import math num = 8.752 result = math.ceil (num) print (result)	This example shows that the 'math.ceil()' subprogram returns the nearest integer not less than the original. The output is: 9
import math num = 8.752 result = math.floor (num) print (result)	The 'math.floor()' subprogram returns nearest integer not greater than the original. The output is: 8
import math num = 25 result = math.sqrt (num) print (result)	This is an example of the square root subprogram. The output is: 5.0
import math print (math.pi)	Pi is a constant, not a subprogram. The pi value is given to fifteen decimal places. The output is: 3.141592653589793 When performing a mathematical calculation using pi, it's best practice to use the pi value given by the math module rather than hard coding the value.

Time library module

The 'time.sleep()' function suspends program execution for a given number of seconds, after which executions resumes at the next line of the program.

This table shows an example of using the time library module subprogram.

Example	Description
import time print ("Sleeping ...") time.sleep (5) print ("Awake")	This example prints the sleeping message, waits 5 seconds, then prints the awake message. The output is: Sleeping ... Awake

String subprograms

This table shows examples of using the string subprograms.

Example	Description
str_one = "hello" length = len (str_one) print (length)	This example finds the length of a string. The output is: 5
str_one = "hello" where = str_one.find ("ell") print (where)	This example finds the start of the substring, inside the original string. It will return -1, if the substring is not there. The output is: 1
str_one = "hello" where = str_one.index ("o") print (where)	This example finds the starting index of the substring, inside the original string. It will report an error if the substring is not there. The output is: 4
status = str_one.isalpha () print (status)	This example checks to see if the string is all alphabetic characters. The output is: True
str_two = "123XYZ" status = str_two.isalnum () print (status)	This example checks to see if the string is all alphabetic and numeric characters. The output is: True
str_three = "789" status = str_three.isdigit () print (status)	This example check to see if the string is all digits. The output is: True
str_one = "Hello world" str_two = "Sun" str_three = str_one.replace ("world", str_two) print (str_three)	This example replaces a substring in the original string, with a new substring. The output is: Hello Sun
str_one = "cat,dog,fox" str_list = str_one.split (",") print (str_list)	This example splits a string into a list, based on the character supplied, in this case a ','. The output is: ['cat', 'dog', 'fox']
str_one = "*ABC**" str_two = str_one.strip ("*") print (str_two)	This example removes all occurrences of a character from the front and back of the original string. The output is: ABC
str_one = "abc" str_two = str_one.upper () print (str_two)	This example converts the original string to all uppercase. The output is: ABC
str_one = "XYZ" str_two = str_one.lower () print (str_two)	This example converts the original string to all lowercase. The output is: xyz
str_one = "ABCxyz" status = str_one.isupper () print (status)	This example checks to see if a string is all uppercase characters. The output is: False
str_one = "abcxyz" status = str_one.islower () print (status)	This example checks to see if a string is all lowercase characters. The output is: True