

Functions

Defining a Function

```
def add(a, b):
    """
    adds two numbers
    """
    z = a + b
    return z
```

- general syntax:
 - (1) `def` statement (name and parameters)
 - (2) `docstring`
 - (3) statements to compute
 - (4) (optional) return statement

```
>>> x = add(10, 15)
```

```
>>> x
```

15

Functions as First Class Objects

- functions can be passed as arguments
- can be used as any other value

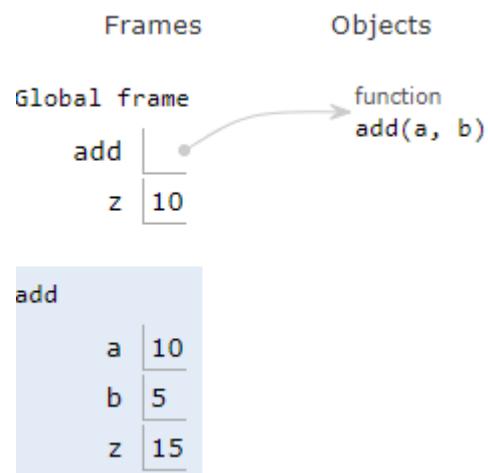
```
def add(a, b):  
    return a + b
```

```
>>> print(add)  
<function add at 0x7f1566d8cde8>  
>>> print(add(3,5))  
8  
>>> print(type(add))  
<type 'function'>
```

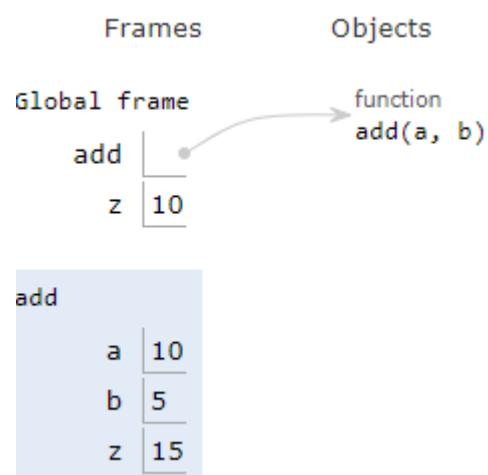
Function and Variable Scope

```
def add(a, b):  
    z = a + b  
    return z
```

```
>>> z = 10; z = add(z, 5)
```



>>> z

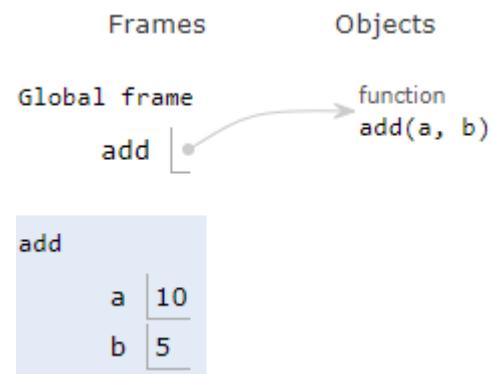


Functions With Positional Parameters

```
def add(a, b):  
    z = a + b  
    return z
```

```
>>> res = add(10, 5)
```

- parameters are bound by position



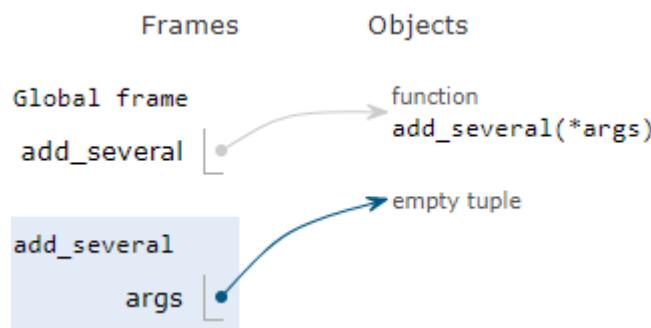
```
>>> res
```

15

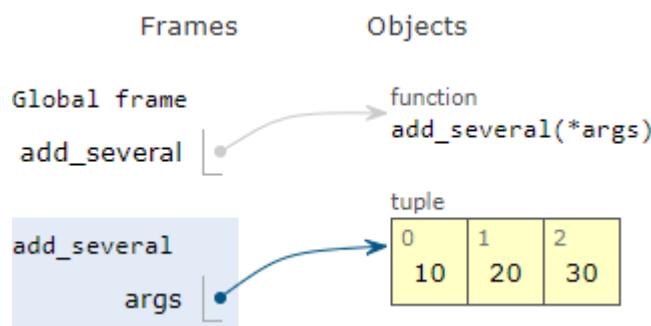
Variable Positional Args

```
def add_several(*args):
    result = 0
    for x in args:
        result = result + x
    return result
```

>>> add_several()



>>> add_several(10, 20, 40)

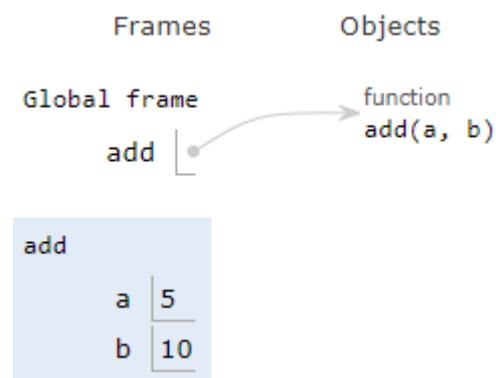


Optional Parameters

```
def add(a, b = 10):  
    z = a + b  
    return z
```

```
>>> add(5)
```

15



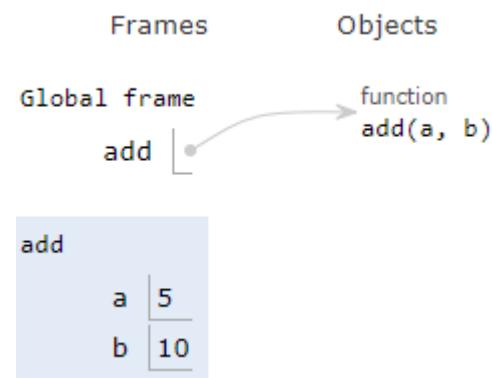
- default is bound to the function upon creation (not when it is called)

Keyword Arguments

```
def add(a, b):  
    z = a + b  
    return z
```

```
>>> res = add(b=10, a=5)
```

- parameters are bound by name



```
>>> res
```

15

Parameter Passing

- **call by value:**

1. function copies values of parameters
2. any changes to parameters are not visible to calling function

- **call by reference:**

1. function uses address of parameters
2. any changes to parameters are visible

- Python uses **call by assignment**

Call by Value Parameter Passing

- call by value:
 1. function copies values of parameters
 2. any changes to parameters are not visible to calling function

```
def modify(data):  
    data.append('Mozzarella')
```

```
>>> x = [1, 2, 3]  
>>> modify(x)  
>>> print('received x = ', x)
```

- call by value should return
received x = [1, 2, 3]

Call by Reference Parameter Passing

- call by reference:
 1. function uses address of parameters
 2. any changes to parameters are visible

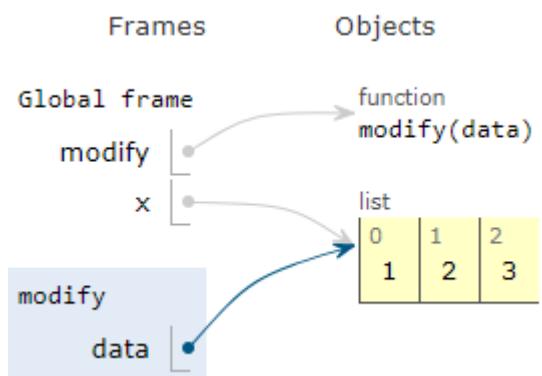
```
def modify(data):
    data = 'Brie'
```

```
>>> x = 'Mozzarella'
>>> modify(x)
>>> print('received x = ', x)
```

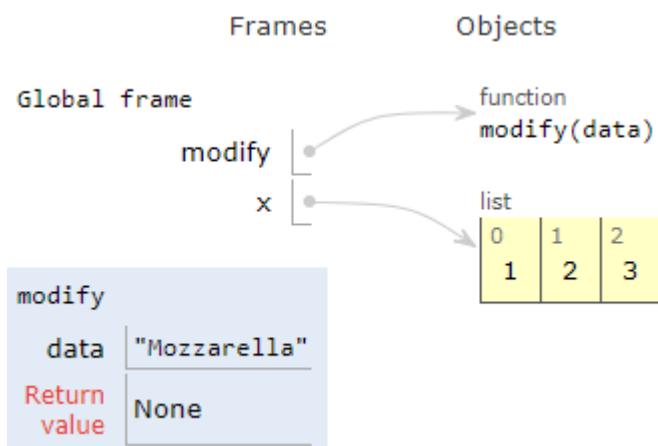
- call by reference should return:
received x = 'Brie'

Mutable Objects with Rebinding (in detail)

```
>>> x = [1, 2, 3]
```



```
>>> modify(x)
```



```
>>> print('received x = ', x)
```

received x = [1, 2, 3]

Parameter Passing: Mutable Objects without Rebinding

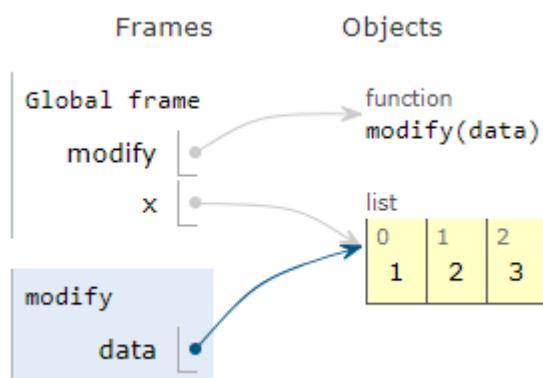
- called function gets a reference to the object and can change it
- calling function points to mutated object without rebinding – changes are visible

```
def modify(data):  
    data.append('Mozzarella')
```

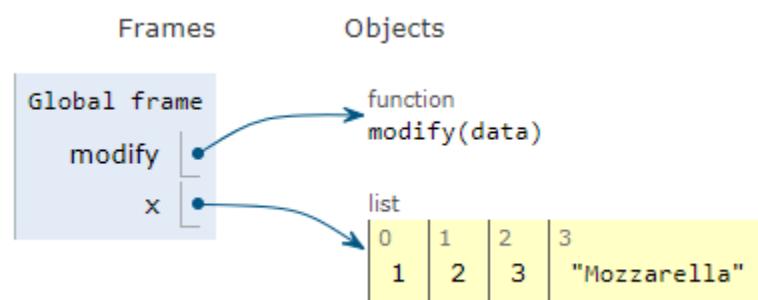
```
>>> x = [1, 2, 3]  
>>> modify(x)  
>>> print('received x = ', x)  
received x = [1, 2, 3, 'Mozzarella']
```

Mutable Objects without Rebinding (details)

- data points to the same object



- modify object



```
>>> print('received x = ', x)  
received x = [1, 2, 3, 'Mozzarella']
```

Passing Immutable Objects as Parameters

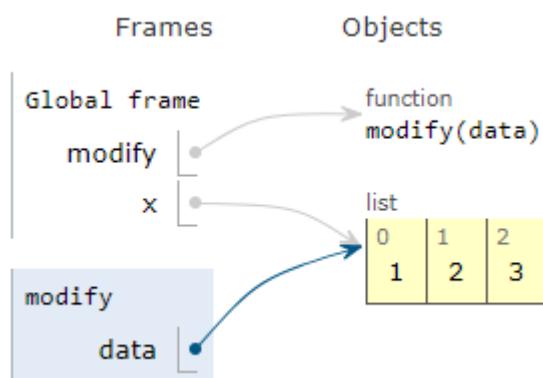
- called function gets a reference to the object and cannot change it
- calling function points to same object: no changes to the object are possible

```
def modify(data):  
    data = 'Brie'
```

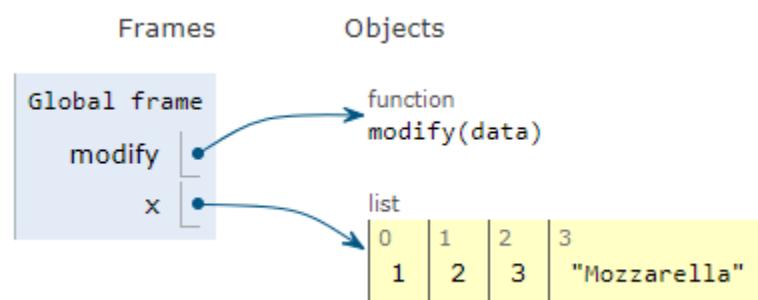
```
>>> x = 'Mozzarella'  
>>> modify(x)  
>>> print('received x = ', x)  
received x = 'Mozzarella'
```

Mutable Objects without Rebinding (details)

- data points to the same object



- modify object



```
>>> print('received x = ', x)  
received x = [1, 2, 3, 'Mozzarella']
```

Parameter Passing: Mutable Objects with Rebinding

- called function gets a reference to the object and can change it
- calling function points to mutated object with rebinding – changes are invisible

```
def modify(data):  
    data = ['Mozzarella']
```

```
>>> x = [1, 2, 3]  
>>> modify(x)  
>>> print('received x = ', x)  
received x = [1, 2, 3, 'Mozzarella']
```

Parameter Passing Summary

- arguments in functions are objects
- all variables reference an Object
- arguments are passed by reference
- mutable arguments can be changed in a function — changes are visible outside
- changes to immutable arguments result in new object(s) creation — changes are not visible outside

yield Statement

- suspends function execution
- produces a sequence of values to caller
- retains enough state to resume
- use `yield` to iterate over a sequence
- if function body contains `yield`, function becomes a generator

yield Example

```
def compute_next_cube():

    x = 1

    while True:

        yield x**3

        x = x + 1
```

- code to test

```
for result in compute_next_cube():

    if result > 1000:

        break

    print result
```

- prints

1, 8, 27 ,64, 125, 216, 343, 512, 729, 1000

Review Problems

Interview Problem

- why are functions considered first class objects in Python?

Interview Problem

- how do you create a Python function?

Interview Problem

- how does a function return values?

Interview Problem

- what happens when a function does not have a return statement?
- is this valid?

Interview Problem

- what will be the output of the following?

```
def func(x,*y,**z):  
    print z
```

```
>>> func(1,2,3)
```

Interview Problem

- what does this code output?

```
def f(x, l=[]):  
    for i in range(x):  
        l.append(i*i)  
    print(l)
```

Interview Problem

- what does `*args` and `**kwargs` mean?

Interview Problem

- are function parameters passed by value or by reference?

Interview Problem

- differentiate local and global variables

Interview Problem

- what are decorators?

Interview Problem

- what does the *yield* keyword do?

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below the navigation bar, the title 'Exercise 51008 —' is displayed. Underneath the title, there are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located below the tabs. The main content area contains the following text:

Define a function `is_prime` that receives an integer argument and returns true if the argument is a prime number and otherwise returns false. (An integer is prime if it is greater than 1 and cannot be divided evenly [with no remainder] other than by itself and one. For example, 15 is not prime because it can be divided by 3 or 5 with no remainder. 13 is prime because only 1 and 13 divide it with no remainder.) This function may be written with a for loop, a while loop or using recursion.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons labeled 'PREV' and 'NEXT' with circular arrows, followed by the word 'Workbench'. To the right are help and settings icons. Below this, the title 'Exercise 51009 —' is displayed. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located below the tabs. The main content area contains the following text:

Assume the availability of a function `is_prime`. Assume a variable `n` has been associated with positive integer. Write the statements needed to compute the sum of the first `n` prime numbers. The sum should be associated with the variable `total`.

Note: `is_prime` takes an integer as a parameter and returns True if and only if that integer is prime.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a software interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and a gear icon on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. The 'WORK AREA' tab is selected. In the main content area, there is a button labeled 'Content Support'. The text of the exercise is as follows:

Assume the availability of a function `is_prime`. Assume a variable `n` has been associated with positive integer. Write the statements needed to find out how many prime numbers (starting with 2 and going in increasing order with successively higher primes [2,3,5,7,11,13,...]) can be added before exceeding `n`. Associate this number with the variable `k`.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon with a gear symbol on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Write the **definition** of a function `typing_speed`, that receives two parameters. The first is the number of words that a person has typed (an int greater than or equal to zero) in a particular time interval. The second is the length of the time interval in seconds (an int greater than zero). The function returns the typing speed of that person in words per minute (a float).

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The image shows a screenshot of a programming exercise interface. At the top, there is a toolbar with green buttons for 'PREV' and 'NEXT', a 'Workbench' button, and a help icon. Below the toolbar, the text 'Exercise 51073 —' is displayed. Underneath this, there are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Write the code to call the function named `send_signal`. There are no parameters for this function.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and gear icon on the right. Below these is the title 'Exercise 51074 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A 'Content Support' button is located above a large text area. The main text area contains the following instructions:

Write the code to call a function whose name is `send_number`. There is **one argument** for this function, which is an int. Send 5 as an argument to the function.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The image shows a screenshot of a programming exercise interface. At the top, there is a toolbar with green buttons for 'PREV' and 'NEXT', a 'Workbench' button, and a help/gear icon. Below the toolbar, the text 'Exercise 51075 —' is displayed. Underneath this, there are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located below the tabs. The main area contains the following text:

Write the code to call a function named send_variable and that expects a single int parameter.

Suppose a variable called x refers to an int. Pass this variable as an argument to send_variable.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button in blue and red, and a help/gear icon. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located above a large text area. The main text area contains the following instruction:

Write the code to call a **function** named `send_two` and that expects **two parameters**: a float and an int. Invoke this function with `15.955` and `133` as **arguments**.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a software interface for a programming exercise. At the top, there are navigation buttons labeled "PREV" and "NEXT", a title "Workbench", and icons for help and settings. Below the title, it says "Exercise 51077 –". There are two tabs: "WORK AREA" (which is selected) and "SOLUTIONS". A "Content Support" button is located above a large text area. The main text area contains the following instructions:

Write the code to call a **function** named `send_object` and that expects **one parameter**, of type `Customer`.

Suppose there is an **object** of type `Customer`, referred to by `John_Doe`. Use this object as an argument to the function.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and help/score icons on the right. Below the navigation is the title 'Exercise 51105'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Write a **function** `min` that has **three** str parameters and returns the **smallest**. (Smallest in the sense of coming first alphabetically, not in the sense of "shortest".)

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below the navigation is the title 'Exercise 51143 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located below the tabs. The main content area contains the following text:

Assume that `print_todays_date` is a **function** that uses no parameters and returns no value. Write a **statement** that calls (invokes) this **function**.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' in green rounded rectangles, followed by a 'Workbench' button in blue, and a help icon ('?') and settings icon ('⚙️'). Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located above a scrollable area containing the exercise instructions.

Exercise 51144 –

WORK AREA **SOLUTIONS**

Content Support

Assume that `print_error_description` is a **function** that expects one **int** parameter and returns no value.

Write a **statement** that invokes the **function** `print_error_description`, passing it the **value 14**.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and gear icon on the right. Below these is the title 'Exercise 51145 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Given `print_larger`, a **function** that expects two parameters and returns no value and given two **variables**, `sales1` and `sales2`, that have already been defined, write a **statement** that calls `print_larger`, passing it `sales1` and `sales2`.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a software interface for a programming exercise. At the top, there are navigation buttons labeled "PREV" and "NEXT", a title "Workbench", and help/symbol buttons. Below the title, the text "Exercise 51146 —" is displayed. A tab bar includes "WORK AREA" and "SOLUTIONS". A "Content Support" button is located in the upper right of the main area. The main content area contains two parts of text:

Given that `add`, a **function** that expects two `int` parameters and returns their sum, and given that two **variables**, `euro_sales` and `asia_sales`, have already been defined:

Write a **statement** that calls `add` to compute the sum of `euro_sales` and `asia_sales` and that associates this value with a **variable** named `eurasia_sales`.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there is a toolbar with green buttons for 'PREV' and 'NEXT', a 'Workbench' tab, and a help icon. Below the toolbar, the title 'Exercise 51147 —' is displayed. The interface features two tabs: 'WORK AREA' and 'SOLUTIONS'. A 'Content Support' button is located in the top right of the main content area. The main content area contains the following text:

Assume that `to_the_power_of` is a **function** that expects two `int` parameters and returns the value of the first parameter raised to the power of the second parameter.

Write a **statement** that calls `to_the_power_of` to compute the value of `cube_side` raised to the power of 3 and that associates this value with `cube_volume`.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a software interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon with a question mark and a gear icon on the right. Below the navigation is the title 'Exercise 51148 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

max is a **function** that expects two int parameters and returns the value of the larger one.

Two variables, population1 and population2, have already been defined and associated with int values.

Write an **expression** (not a statement!) whose value is the larger of population1 and population2 by calling max.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below the navigation is the title 'Exercise 51149 —'. Underneath the title are two tabs: 'WORK AREA' (selected) and 'SOLUTIONS'. A green button labeled 'Content Support' is located below the tabs. The main content area contains three paragraphs of text:

Assume that `max2` is a **function** that expects two `int` parameters and returns the value of the larger one.

Also assume that four variables, `population1`, `population2`, `population3`, and `population4` have already been defined and associated with `int` values.

Write an **expression** (not a statement!) whose value is the largest of `population1`, `population2`, `population3`, and `population4` by calling `max2`.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('⚙️') on the right. Below these are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is visible above a large text area. The main text area contains the following instruction:

Write the **definition** of a function **twice**, that receives an **int** parameter and returns an **int** that is **twice** the value of the parameter.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('gear') on the right. Below these are two tabs: 'WORK AREA' (highlighted in green) and 'SOLUTIONS'. A 'Content Support' button is located below the tabs. The main area contains the following text:

Exercise 51156 —

Write the **definition** of a function **add**, that receives two int parameters and returns their **sum**.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a question mark icon and gear icon on the right. Below these is the title 'Exercise 51158 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS'. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Write the **definition** of a function `power_to`, which receives two parameters. The first is a double and the second is an int. The function returns a double.

If the second parameter is negative, the function returns zero. Otherwise it returns the value of the first parameter raised to the power of the second.

Programming Exercise

Worksheet

this page is intentionally left blank

Programming Exercise

The screenshot shows a user interface for a programming exercise. At the top, there are navigation buttons: 'PREV' and 'NEXT' on the left, 'Workbench' in the center, and a help icon ('?') and settings icon ('gear') on the right. Below the navigation is the title 'Exercise 51218 —'. Underneath the title are two tabs: 'WORK AREA' and 'SOLUTIONS', with 'WORK AREA' being the active tab. A green button labeled 'Content Support' is located above the main content area. The main content area contains the following text:

Write the **definition** of a function `max` that has three `int` parameters and returns the **largest**.

Programming Exercise

Worksheet

this page is intentionally left blank