ISYS90088
Introduction to Application Development

Week 10 – Contd. from week 9 on functions

Semester 2 , 2018

Dr Antonette Mendoza

s

Recap: Functions and return statement

A value-returning function has a return statement that returns a value back to the part of the program that called it.

Assignment Project Exam Help

Syntax:

```
def <function_name>():
    statement
    statement
    statement
    return <expression>
```

https://powcoder.com

Add WeChat powcoder

Functions and return statement

- The value of the expression that follows the key word return will be sent back to part of the program that called this function. This can be any value, expression, or variable that has a value.

- A return statement can also send back:
  - Strings
  - boolean values
  - multiple values

(recap)Examples 1: return statement

write a program that converts cenlius to farenheit

```
def C2F(n):
    return 9*n/5 + 32


def main():
    cels = int(input('enter a value in celcius:'))
    f = C2F(cels)
    print(f)

main()
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

(recap) Examples 2: return statement

```python
def main():
    # get the user's age
    first_age = int(input('enter your age:'))
    # get the users best friends age
    second_age = int(input('enter your best friends age:'))
    # get the sum of both ages
    total = sum (first_age, second_age)
    # display the total age
    print('their total age is:', total, 'years old')

#sum function accepts two int arguments & returns sum of those
arguments
def sum(num1, num2):
    result = num1 + num2
    return result
#call main function
main()
```

Returning strings

```python
def get_name():
    name = input('enter your name:')
    return name


def main():
    print('this example prints a name given by user:')

    user_name = get_name()
    print('my name is' , user_name)


main()
```

Quiz!

- What is printed to the screen here?

```python
def bloodify1(word):
    return word[:3] + '-bloody-' + word[3:]

print(bloodify1("fantastic"))
```

Returning boolean values

- You receive an integer from the user. Write a function that checks whether or not this integer is even or odd and returns a boolean.

➤ What are the tasks in this program?

- accept value from user (main function)

- checks whether the value is even   (funct to check)

  or odd and returns a boolean

- print back a response (main function)

- and don't forget to call the main!!!

Example: Returning boolean values

* fill relevant statements in the place marked XXX

```python
def main():
    number = int(input('enter a number:'))
    if XXX:
        print('the number is even')
    else:
        print('the number is odd')


# how do you check if a number is even or odd

def is_even(num1):
    if (XXX) == 0:
        status = True
    else:
        status = False
    return XXX

main()
```

Returning multiple values

Syntax:

```
 return  expr1, expr2, expr..etc…
```

Example:

#a function to accept first and last name and then print out your full name

```
    def get_name():
        first = input('enter your first name:')
        last = input('enter your last name:')
        return first, last

    def main():
        first_name, last_name = get_name()
        print('My name is:', first_name, last_name)

    main()
```

Returning multiple values: tuples

```
def checking_tuple(x):
  sum = x + 1
  mult = x * 3
  exp = x ** 3
  return (sum, mult, exp)


def main():
    num = 5
    (a, b, c) = checking_tuple(num)
    print(a,b,c)
main()
```

Key arguments

- While generally , arguments are passed by position to parameter variables in functions, you can also specify which parameter variable the argument should be passed to.

Syntax:

```
parameter_name = value
```

In this format, `parameter_name` is the parameter and `value` is the value being passed to that parameter.

An argument written in this format is called a key argument.

Key arguments: examples

```
def main():
    show_interest(rate = 0.01, periods = 10,
principal = 100000.0)



def show_interest(principal, rate, periods):
    interest = principal * rate * periods
    print('The simple interest will be $%.2f' %
interest)



main()
```

Mixing Key arguments with positional arguments : examples

- You can mix keyword arguments with positional arguments. Positional arguments must come first followed by keyword arguments

```python
def main():
    show_interest(10000.0, periods = 20, rate = 0.01)


def show_interest(principal, rate, periods):
    interest = principal * rate * periods
    print('The simple interest will be $%.2f' %
interest)


main()
```

Functions and Processing lists – passing lists as arguments

- Write a function that calculates the total of the values in a list:

```
#function to calculate the total in a list of numbers
def main():
    #create a list
    numbers = [1,2,3,4,5,6]
    print('the total is ', get_total(numbers))


def get_total(list1):
    # create an accumulator
    total = 0
    for num in list1:
        total = total + num
    return total


main()
```

Functions and Processing lists – passing lists as arguments

- Write a function that calculates the average of the values in a list and returns the average and the total:

```
def main():
    #create a list
    numbers = [1,2,3,4,5,6]
    total = 0
    print('the total is', get_average(numbers, total))


def get_average(list1,t):
    # create an accumulator
    total = 0.0
    for num in list1:
        total = total + num
        average = total/len(list1)
    return average, total

main()
```

Functions & Processing lists – returning a list

- A function can return a reference to a list.

- For example, you might create a list, add items into it and then return a reference to the list so that parts of the program can work on it.

```python
def main():
    # get a list with values stored in it
    numbers = get_values()
    print('the numbers in the list are:',
numbers)
```

Functions & Processing lists – returning a list

```python
def get_values():
    #create an empty list
    values = []
    #create a variable to control the loop
    again = 'y'
    # get values from the user and add into the list
    while again == 'y':
        num = int(input('enter a number:'))
        values.append(num)
        #to add more items in the list
        print('do you want to enter another number?')
        again = input('y = yes, anything else = no:')
        print()
    return values
main()
```

Functions and Processing lists – passing lists as arguments – try this!

- Write a program that uses two function that gets a series of test scores and calculates the average of the scores with the lowest score dropped:

```python
def main():
    #create a list of scores
    scores = get_values()
    # calculates the total of the list of elements
    total = get_total(scores)
    lowest = min(scores)
    #subtract the lowest from the total
    total = total - lowest
    average = total/(len(scores) - 1)
    print('the average is, with the lowest dropped   out:', average)

main()
# see example code for the functions get_values and get_total
```

Functions and Processing dictionaries – passing dictionaries as arguments

Tasks:

- Look up a dictionary
- Add items into a dictionary
- Change items in the dictionary
- Delete items

Functions and Processing dictionaries – passing dictionaries as arguments

- **Look up a dictionary**

```
birthday = {'chris':'15-03-1978', 'matty': '03-03-189',
'tom': '02-10-2000'}
```

```
# look up a dictionary
def look_up(b):
    #name to look up
    name = input('enter a name:')
    #print(birthday.get(name, 'not found'))
    #or can use a if loop
    if name not in b.keys():
        print('not found')
    else:
        print('found')

look_up(birthday)
```

Functions and Processing dictionaries – passing dictionaries as arguments

- **Add items into a dictionary**

```python
birthday = {'chris':'15-03-1978', 'matty': '03-03-1998',
'tom': '02-10-2000'}

# add a new birthday
def add_birthday(b):
    #get the name and birthday
    name = input('enter the name:')
    bday = input('enter the bday:')
    if name not in b.keys():
        b[name] = bday #add value to the new key
        print(b)
    else:
        print('entry exists')

add_birthday(birthday)
```

Functions and Processing dictionaries – passing dictionaries as arguments

- **Change items in a dictionary**

```
birthday = {'chris':'15-03-1978', 'matty': '03-03-1998',
'tom': '02-10-2000'}
def change_value(b):
    name = XXX
    if XXX in XXX:
        cbday = XXX
        XXX = XXX
        print(b)
    else:
        print('not there')

# main function
XXX
```

Functions and Processing dictionaries – passing dictionaries as arguments

- **Change items in a dictionary**

```
birthday = {'chris':'15-03-1978', 'matty':
'03-03-1998', 'tom': '02-10-2000'}
def change_value(b):
    name = input('enter the name:')
    if name in b:
        cbday = input('enter the bday:')
        b[name] = cbday
        print(b)
    else:
        print('not there')


# main function
change_value(birthday)
```

Another example!

```
# example – send in details into a function that creates
#a dictionary and returns it back to the main


def contact(personName, mobile):
    mobileContact = {'name':personName, 'mobile':mobile}
    return (mobileContact)

myfriend = contact('Alice', '04231234')
print(myfriend)
```

Finally!

- Check out all examples on the LMS

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder