# Worksheet 3b Functions Answers

**Mars Probe Lost Due To Simple Math Error**

NASA lost its $125-million Mars Climate Orbiter because spacecraft engineers failed to convert from English to metric measurements when exchanging vital data before the craft was launched, space agency officials said Thursday.

A navigation team at the Jet Propulsion Laboratory used the metric system of millimeters and meters in its calculations, while Lockheed Martin Astronautics in Denver, which designed and built the spacecraft, provided crucial acceleration data in the English system of inches, feet and pounds.

As a result, JPL engineers mistook acceleration readings measured in English units of pound-seconds for a metric measure of force called newton-seconds.

In a sense, the spacecraft was lost in translation.

"That is so dumb," said John Logsdon, director of George Washington University's space policy institute.

*Source: R Hotz, LA Times (http://articles.latimes.com/1999/oct/01/news/mn-17288)*

1. **Conversion Table**

Use the conversion table to create a range of functions.

Each function should complete one possible conversion.

|  |  |
| --- | --- |
| **One** | **Equals** |
| 1 inch | 2.54 cm |
| 1 mile | 1.6093 km |
| 1 foot | 0.3048 m |
| 1 yard | 0.9144 m |
| 1 gallon | 4.546 l |
| 1 pound | 0.454 kg |
| 1 ounce | * 1. g |

Test each function by calling it at the bottom of the program.

#Program L3 WS3b Ex1.py

def milesToKm():  
 miles = int(input(“How many miles? ”))

km = miles \* 1.6093

print(miles, “miles =”, km, “km”)  
  
def feetToM():

feet = int(input(“How many feet? ”))

metres = feet \* 0.3048

print(feet, “ft =”, metres, “m”)  
# MAIN PROGRAM

milesToKm()  
feetToM()

***(See program L3 WS3b Ex1.py in Sample programs folder)***

**Extension:**

Add a menu that will prompt the user to select a conversion option.  
The program then calls call the relevant conversion function.

#Program L3 WS3b Ex1 extension.py

def milesToKm():  
 miles = int(input(“How many miles? ”))

km = miles \* 1.6093

print(miles, “miles =”, km, “km”)  
  
def feetToM():

feet = int(input(“How many feet? ”))

metres = feet \* 0.3048

print(feet, “ft =”, metres, “m”)

# MAIN PROGRAM

print(“MAIN MENU”)

print(“1. Convert miles to km”)  
print(“2. Convert feet to metres”)  
choice = int(input(“Enter a choice: “))  
if choice == 1:

milesToKm()

elif choice == 2:  
 feetToM()

else:

print(“Error, invalid”)

A more advanced program could have this menu inside a subroutine of its own and and could be called in a loop

def menu():

print(“MAIN MENU”)

print(“1. Convert miles to km”)  
print(“2. Convert feet to metres”)

print(“0. Quit”)  
choice = int(input(“Enter a choice: “))  
if choice == 0:

exit()

elif choice == 1:

milesToKm()

elif choice == 2:  
 feetToM()

else:

print(“Error, invalid”)

# MAIN MENU

while True:

menu()

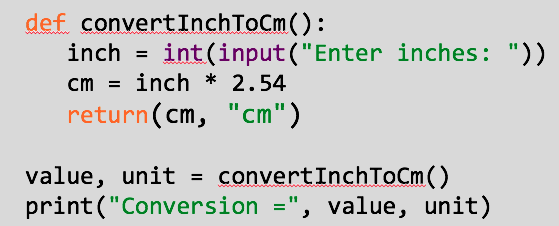
***(See program L3 WS3b Ex1 extension.py in Sample programs folder)***

1. **Functions**

Change your conversion program from Task 1 so that each subroutine is now a function that returns a value and a unit.

The main program (at the bottom of the page) should then display the result instead of the subroutine doing so.

e.g.



#Program L3 WS3b Ex2.py

def milesToKm():  
 miles = int(input(“How many miles? ”))

km = miles \* 1.6093

return km, “km”  
  
def feetToM():

feet = int(input(“How many feet? ”))

metres = feet \* 0.3048

return metres, “m”  
  
# MAIN PROGRAM

value, unit = milesToKm()

print(“Conversion =”, value, unit)

value, unit = feetToM()

print(“Conversion =”, value, unit)

***(See program L3 WS3b Ex2.py in Sample programs folder)***

1. **Parameter Passing**

Write a program that will ask for the radius of a circle and then calculate and display that circle’s circumference (2 x 3.14 x radius) and that circle’s area (2 x 3.14 x 3.14).

The program should be made up of 4 subroutines – getRadius(), calculateCircumference(), calculateArea() and displayResults().

You should make sure that you pass parameters where necessary to make the program work fully.

#Program L3 WS3b Ex3.py

def getRadius():  
 radius = int(input(“Enter radius of circle: ”))

return radius  
  
def calculateCircumference(radius, pi):

circ = 2 \* pi \* radius

return circ

def calculateArea(radius, pi):

area = pi \* radius \* radius

return area

def displayResults(radius, circ, area):

print(“A circle of radius”, radius)

print(“Has a circumference of”, circ)

print(“And an area of”, area)

# MAIN PROGRAM

pi = 3.14

radius = getRadius()

circ = calculateCircumference(radius, pi)

area = calculateArea(radius, pi)

displayResults(radius, circ, area)

4. Write similar programs for calculating the volume and total surface area of a cuboid, triangular prism and closed cylinder – using decomposition to identify separate subroutines and make sure to use parameter passing.

#Program L3 WS3b Ex4 cuboid.py

def getDimensions():  
 height = int(input(“Enter height of the cuboid: ”))

width = int(input(“Enter width of the cuboid: ”))

depth = int(input(“Enter depth of the cuboid: ”))

return height, width, depth  
  
def calculateArea(height, width, depth):

side1 = height \* width

side2 = width \* depth

side3 = height \* depth

totalArea = 2 \* (side1 + side2 + side3)

return totalArea

def calculateVol(height, width, depth):

volume = height \* width \* depth

return volume

def displayResults(area, volume):

print(“The cuboid has a total surface area of”, area)

print(“And a volume of”, volume)

# MAIN PROGRAM

height, width, depth = getDimensions()

area = calculateArea(height, width, depth)

volume = calculateVol(height, width, depth)

displayResults(area, volume)

***(See Program L3 WS3b Ex4.py in Sample programs folder)***