**Function Basics**

Możemy tworzyć funkcje w Pythonie, korzystając z następujących:

• Słowo kluczowe def

• Nazwa funkcji - małe litery zaczynające się od litery lub podkreślenia (\_)

• Lewy nawias (()

• 0 lub więcej nazw argumentów

• Prawy nawias ())

• Dwukropek:

• Zdefiniowane ciało funkcji

Oto przykład bez argumentu:

>>> def hello\_world():

... print("Hello, World!")

...

>>> hello\_world()

Hello, World!

>>>

Jeśli chcemy zdefiniować argument, umieścimy w nawiasie nazwę zmiennej, którą chcemy, aby miała

>>> def print\_name(name):

... print(f"Name is {name}")

...

>>> print\_name("Keith")

Name is Keith

Spróbujmy przypisać wartość z print\_name do zmiennej:

>>> output = print\_name("Keith")

Name is Keith

>>> output

>>>

Żaden z tych przykładów nie ma wartości zwracanej, ale zazwyczaj chcemy uzyskać wartość zwracaną, chyba że funkcja jest naszą „główną” funkcją lub wykonuje „efekt uboczny”, taki jak drukowanie. Jeśli nie zadeklarujemy jawnie wartości zwracanej, wynikiem będzie None.

Możemy zadeklarować, co powracamy z funkcji używającej słowa kluczowego return:

>>> def add\_two(num):

... return num + 2

...

>>> result = add\_two(2)

>>> result

4

#### Encapsulating Behavior with Functions

Aby zagłębić się w funkcje, napiszemy skrypt, który poprosi użytkownika o pewne informacje i obliczy wskaźnik masy ciała użytkownika (BMI). To nie jest powszechny problem, ale jest to coś, co ma sens jako funkcja i nie wymaga od nas korzystania z funkcji językowych, których jeszcze się nie nauczyliśmy.

Oto wzór BMI:

BMI = (weight in kg / height in meters squared )

W przypadku systemów imperialnych jest to ta sama formuła, z wyjątkiem tego, że wynik należy pomnożyć przez 703.

Chcemy poprosić użytkownika o informacje, zebrać wyniki i dokonać obliczeń, jeśli możemy. Jeśli nie możemy zrozumieć systemu pomiarowego, musimy ponownie poprosić użytkownika o wyjaśnienie błędu.

#### Gathering Info

Ponieważ chcemy mieć możliwość wielokrotnego monitowania użytkownika, zapakujemy nasze połączenia do pojedynczej funkcji, która zwraca krotkę z danymi użytkownika:

def gather\_info():

height = float(input("What is your height? (inches or meters) "))

weight = float(input("What is your weight? (pounds or kilograms) "))

system = input("Are your measurements in metric or imperial units? ").lower().strip()

return (height, weight, system)

Przekształcamy wysokość i wagę w wartości zmiennoprzecinkowe i możemy zgodzić się z potencjalnym błędem, jeśli użytkownik wprowadzi nieprawidłowy numer. W przypadku systemu będziemy standaryzować rzeczy, wywołując niższe, aby zamieniać małe litery, a następnie wywołując pasek, aby usunąć białe znaki od początku i końca.The most important thing about this function is instrukcja return, którą dodaliśmy, aby upewnić się, że możemy przekazać wysokość, wagę i system z powrotem do wywołującego funkcję.

#### Calculating and Printing the BMI

Po zebraniu informacji musimy użyć tych informacji do obliczenia BMI. Napiszmy funkcję, która może to zrobić:

def calculate\_bmi(weight, height, system='metric'):

"""

Return the Body Mass Index (BMI) for the

given weight, height, and measurement system.

"""

if system == 'metric':

bmi = (weight / (height \*\* 2))

else:

bmi = 703 \* (weight / (height \*\* 2))

return bmi

Ta funkcja zwróci obliczoną wartość i możemy zdecydować, co z nią zrobić w normalnym przepływie naszego skryptu.

Trójdzielny ciąg znaków, który użyliśmy na górze naszej funkcji, jest znany jako „ciąg dokumentacji” lub „ciąg dokumentów” i może być użyty do automatycznego generowania dokumentacji naszego kodu za pomocą narzędzi w ekosystemie Pythona.

#### Setting Up The Script’s Flow

Nasze funkcje nie dają nam nic dobrego, jeśli nie zadzwonimy do nich. Teraz nadszedł czas, aby skonfigurować przepływ skryptów. Chcemy być w stanie ponownie nakłonić użytkownika, więc chcemy wykorzystać celową nieskończoną pętlę, z której możemy się wyrwać. W zależności od systemu ustalimy, w jaki sposób powinniśmy obliczyć BMI lub zapytać użytkownika ponownie. Oto nasz przepływ:

while True:

height, weight, system = gather\_info()

if system.startswith('i'):

bmi = calculate\_bmi(weight, system='imperial', height=height)

print(f"Your BMI is {bmi}")

break

elif system.startswith('m'):

bmi = calculate\_bmi(weight, height)

print(f"Your BMI is {bmi}")

break

else:

print("Error: Unknown measurement system. Please use imperial or metric.")

#### Full Script

Po napisaniu naszego skryptu będziemy musieli go uruchomić (używając chmod u + x ~ / bin / bmi).

~/bin/bmi

#!/usr/bin/env python3.6

def gather\_info():

height = float(input("What is your height? (inches or meters) "))

weight = float(input("What is your weight? (pounds or kilograms) "))

system = input("Are your mearsurements in metric or imperial systems? ").lower().strip()

return (height, weight, system)

def calculate\_bmi(weight, height, system='metric'):

if system == 'metric':

bmi = (weight / (height \*\* 2))

else:

bmi = 703 \* (weight / (height \*\* 2))

return bmi

while True:

height, weight, system = gather\_info()

if system.startswith('i'):

bmi = calculate\_bmi(weight, system='imperial', height=height)

print(f"Your BMI is {bmi}")

break

elif system.startswith('m'):

bmi = calculate\_bmi(weight, height)

print(f"Your BMI is {bmi}")

break

else:

print("Error: Unknown measurement system. Please use imperial or metric.")