Input and Output formatting and calculations

Lillebaelt Academy of
Professional Higher Education

Author
Martin Grønholdt
mart80c7@edu.eal.dk

Sunday 6 November 2016

Table of Contents

Introduction

These programs shows how to use basic python input and output functions. The program output has been formatted using Python string formatter, whenever sensible.

6 Sales Tax

This program calculates the county and state sales tax. The program catches input errors by catching the `ValueError` exception. It will output a help messages if this exception occurs.

prog6.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# The above lines tell the shell to use python as interpreter when the
# script is called directly, and that this file uses utf-8 encoding,
# because of the country specific letter in my surname.
'''
Name: Program 6
Author: Martin Bo Kristensen Grønholdt.
Version: 1.0 (6/11-2016)

Calculate the county and state sales tax, and the final total of a purchase.
'''
def main():
    '''
    Program main entry point.
    '''
    # Get the amount of purchase from the user.
    try:
        total_purchase = float(input('Enter the amount of a purchase: '))
    except ValueError:
        # Complain when something unexpected was entered.
        print('\nPlease use only numbers.')
        exit(1)

    # Calculate the county tax.
    county_tax = total_purchase * 0.02
    # Calculate the state tax
    state_tax = total_purchase * 0.04

    # Print the totals
    # Use new style Python 3 format strings.
    # {:12.2f} means align for a total of 12 digits with 2 digits
    # after the decimal point.
    print('\nTotal purchase:\t\t\t{:12.2f}'.format(total_purchase))
    print('State sales tax (4%):\t{:12.2f}'.format(state_tax))
    print('County sales tax (2%):\t{:12.2f}'.format(county_tax))
    print('Final total:\t\t\t{:12.2f}'.format(
        total_purchase + state_tax + county_tax))


# Run this when invoked directly
if __name__ == '__main__':
    main()
```

Result

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog6.py"
Enter the amount of a purchase: 7894

Total purchase:               7894.00
State sales tax (4%):          315.76
County sales tax (2%):         157.88
Final total:                  8367.64
```

Output of the program when run from the command line.

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog6.py"
Enter the amount of a purchase: 2hjjhg

Please use only numbers.
```

Output of the program when the user enters an incorrect value.

7 Miles-per-Gallon

This program uses the entered values for distance and gallons of gas to calculate the miles travelled per gallon. Both input lines are enclosed in the `try...except` block therefore both will trigger a user friendly message if the user inputs something unexpected.

prog7.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# The above lines tell the shell to use python as interpreter when the
# script is called directly, and that this file uses utf-8 encoding,
# because of the country specific letter in my surname.
'''
Name: Program 7
Author: Martin Bo Kristensen Grønholdt.
Version: 1.0 (6/11-2016)

Calculate miles-per-gallon for a given set of miles and gallons.
'''


def main():
    '''
    Program main entry point.
    '''
    # Get the data from the user.
    try:
        miles = float(input('Enter the distance in miles: '))
        gallons = float(input('Enter the gas used in gallons: '))
    except ValueError:
        # Complain when something unexpected was entered.
        print('\nPlease use only numbers.')
        exit(1)

    # Miles per gallon.
    mpg = miles / gallons

    # Print mpg
    # Use new style Python 3 format strings.
    # {:0.2f} means 2 digits after the decimal point.
    print('\n{:0.2f} miles using'.format(miles),
          '{:0.2f} gallons of gas is'.format(gallons),
          '{:0.2f} miles per gallon.'.format(mpg))


# Run this when invoked directly
if __name__ == '__main__':
    main()
```

Result

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog7.py"
Enter the distance in miles: 1024
Enter the gas used in gallons: 10.8

1024.00 miles using 10.80 gallons of gas is 94.81 miles per gallon.
```

Output of the program when run from the command line.

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog7.py"
Enter the distance in miles: 512
Enter the gas used in gallons: twenty

Please use only numbers.
```

Output of the program when given malformed input.

8 Tip, Tax, and Total

This program calculates the tips and sales tax of a restaurant bill.

prog8.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# The above lines tell the shell to use python as interpreter when the
# script is called directly, and that this file uses utf-8 encoding,
# because of the country specific letter in my surname.
'''
Name: Program 8
Author: Martin Bo Kristensen Grønholdt.
Version: 1.0 (6/11-2016)

Calculate the total price of a meal purchased at a restaurant.
'''
def main():
    '''
    Program main entry point.
    '''
    # Get the amount of purchase from the user.
    try:
        total_purchase = float(input('Enter the price of the meal: '))
    except ValueError:
        # Complain when something unexpected was entered.
        print('\nPlease use only numbers.')
        exit(1)

    # Calculate the tip.
    tip = total_purchase * 0.15
    # Calculate the sales tax
    tax = total_purchase * 0.07

    # Print the totals
    # Use new style Python 3 format strings.
    # {:12.2f} means align for a total of 12 digits with 2 digits
    # after the decimal point.
    print('\nTotal purchase:\t\t{:12.2f}'.format(total_purchase))
    print('Tip (15%):\t\t\t{:12.2f}'.format(tip))
    print('Sales tax (7%):\t\t{:12.2f}'.format(tax))
    print('Final total:\t\t{:12.2f}'.format(total_purchase + tip + tax))


# Run this when invoked directly
if __name__ == '__main__':
    main()
```

Result

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog8.py"
Enter the price of the meal: 100.5

Total purchase:               100.50
Tip (15%):                     15.07
Sales tax (7%):                 7.04
Final total:                  122.61
```

Output of the program when run from the command line.

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog8.py"
Enter the price of the meal: FIVEHUNDREDDOLLARS!!!!!!!

Please use only numbers.
```

Output of the program when given malformed input.

9 Celsius to Fahrenheit Temperature Converter

This program converts °C into °F, the output is formatted as a simple line of text.

prog9.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# The above lines tell the shell to use python as interpreter when the
# script is called directly, and that this file uses utf-8 encoding,
# because of the country specific letter in my surname.
'''
Name: Program 9
Author: Martin Bo Kristensen Grønholdt.
Version: 1.0 (6/11-2016)

Convert Celsius temperatures to Fahrenheit temperatures.
'''


def main():
    '''
    Program main entry point.
    '''
    # Get the amount of purchase from the user.
    try:
        celcius = float(input('Enter temperature in degrees Celsius: '))
    except ValueError:
        # Complain when something unexpected was entered.
        print('\nPlease use only numbers.')
        exit(1)

    # Calculate the county tax.
    fahrenheit = (9 / 5) * celcius + 32

    # Print the totals
    # Use new style Python 3 format strings.
    # {:0.2f} means 2 digits after the decimal point.
    print('\n{:0.2f} degrees Celcius is'.format(celcius),
          '{:0.2f} degrees Fahrenheit.'.format(fahrenheit))


# Run this when invoked directly
if __name__ == '__main__':
    main()
```

Result

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog9.py"
Enter temperature in degrees Celsius: -17.78

-17.78 degrees Celcius is -0.00 degrees Fahrenheit.
```

Output of the program when run from the command line.

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog9.py"
Enter temperature in degrees Celsius: cold cold, HOT HOT HOT

Please use only numbers.
```

Output of the program when given malformed input.

10 Stock Transaction Program

This program calculate some data about a stock transaction, it has no input.

prog10.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# The above lines tell the shell to use python as interpreter when the
# script is called directly, and that this file uses utf-8 encoding,
# because of the country specific letter in my surname.
'''
Name: Program 10
Author: Martin Bo Kristensen Grønholdt.
Version: 1.0 (6/11-2016)

Stock Transaction Program.
'''


def main():
    '''
    Program main entry point.
    '''
    # This is the constants for the transaction.
    shares = 1000.0
    commission = 0.02
    bought_price_per_share = 32.87
    sold_price_per_share = 33.92

    # Calculate commissions.
    bought_commission = bought_price_per_share * shares * commission
    sold_commission = sold_price_per_share * shares * commission
    # Get the amount paid for the purchase/sale.
    total_price_bought = bought_commission + bought_price_per_share * shares
    total_price_sold = sold_price_per_share * shares - sold_commission

    #####################################################
    # Output:
    # * The amount of money Joe paid for the stock.
    # * The amount of commission Joe paid his broker when he bought the stock.
    # * The amount that Joe sold the stock for.
    # * The amount of commission Joe paid his broker when he sold the stock.
    # * Display the amount of money that Joe had left when he sold the stock
    #   and paid his broker (both times). If this amount is positive, then Joe
    #   made a profit. If the amount is negative, then Joe lost money.
    #
    # Use new style Python 3 format strings.
    # {:12.2f} means align for a total of 12 digits with 2 digits after the
    # decimal point.
    print('\nTotal stock price when bought:\t\t',
          '{:12.2f}'.format(total_price_bought))
    print('Commission paid when bought:\t\t',
          '{:12.2f}'.format(bought_commission))
    print('Total stock price when sold:\t\t',
            '{:12.2f}'.format(total_price_sold))
    print('Commission paid when sold:\t\t\t',
            '{:12.2f}'.format(sold_commission))
    print('Total win/loss of the transaction:\t',
            '{:12.2f}'.format(total_price_sold - total_price_bought))


# Run this when invoked directly
if __name__ == '__main__':
    main()
```

Result

```
/usr/bin/python3.5 "/mnt/data/Documents/Skole/It-et/2016/programming/Assignent
B1/prog10.py"

Total stock price when bought:           33527.40
Commission paid when bought:               657.40
Total stock price when sold:             33241.60
Commission paid when sold:                 678.40
Total win/loss of the transaction:        -285.80
```

Output of the program when run from the command line.

Conclusion

There is a lot of possibilities for creating nice output using the python formatting functions. There are of course both GUI and TUI libraries for things like QT, GTK and ncurses, but much can be achieved by the formatting functions.