MBSI Coding Workshop Team's Recommended Resources

Courses

1. EdX (**Carl**): free courses on many programming topics offered by top universities. Recommend CS50 for computer science-related knowledge
2. DataCamp (**Carl + Ryan**):
    - *Description* - learn Python, R, SQL and other data science-related skills through interactive courses, practice coding challenges and projects
    - *Why you recommend it* - if you're planning to handle data and got time to do a full course DataCamp is recommended for learning R. Courses have bite-sized video lessons with downloadable lesson slides and hands-on exercises with a really nice user interface. Can assess your skill level and recommend appropriate courses. Courses can also be organised by skill or career track.
    - *Cost* - Free for limited courses, projects and skills assessment. Pay 25 USD/month for all courses, career and skill tracks, assessments, and support. Personally, I just used the free stuff.
3. Kaggle (**Nick**):
    - *Description* - Machine learning community with user-made tutorials, projects, discussions and short courses. Full-on competitions and datasets.
    - *Why you recommend it* - Heaps of really smart people contributing ideas. Free cloud service to run your code on a GPU. A bunch of short courses(3 hours or less) on topics such as data visualisations, data cleaning, pandas as well as more machine learning topics. However, specific to data science and machine learning so if you're wanting to learn programming skills outside of those fields, another website may be more suitable.
    - *Cost* - Free
4. Codecademy (**Ryan**):
    - *Description* - basic interactive courses, practice questions and peer support forums (free). Pay subscription for more courses, extra practice packs and quizzes, step-by-step guidance, certifications, and projects.
    - *Why you recommend it* - available in multiple languages, organises its courses by skill or career path, interactive, and has a large user base.
    - *Cost* - price may be prohibitive, costing 20 USD/month for pro subscription but with >35% discount for students. Personally, I just used the free material.
5. Udemy (**Daniel**):
    - *Description* - learn all sorts of skills from programming to hardware development to drawing. Perfect for when you decide to pick up something new
    - *Why you recommend it* - lifetime access to purchased courses, learn at your own pace, material often taught by industry experts

- *Cost* - Can be pricey (max of 200 dollars), however, there are often discounts (up to 90%)

Problem-based websites

1. PracticePython (**Allen**) (*easy-intermediate*): a range of problems you can try tackling; good opportunity to practise Python basics
2. Rosalind (**Allen**) (*easy-advanced*): bio-related problem set with a wide range of difficulty
3. ProjectEuler (**Allen**) (*intermediate-advanced*): a range of trickier problems; problems often require some basic and intermediate programming tricks, as well as some maths knowledge.
4. HackerRank (**Ryan**) (*easy-advanced*): problem-based learning with discussion board. Problems can be solved in multiple languages. Can also take skills certification assessments and compete in programming contests. Geared towards people who already have a handle on the basics.
5. CodeWars (**Daniel**) (*easy-advanced*): similar to HackerRank but without certifications. You can adjust the difficulty of the problems you get by changing your experience level. 'Spar' with other coders and create your own problems for others to attempt.
6. Leetcode (**Alex**) (*easy-advanced*): a famous coding practice database. It contains a bunch of different levels of coding questions for several popular languages. A good source for the Officers to get sample questions or practical questions for worksheets.

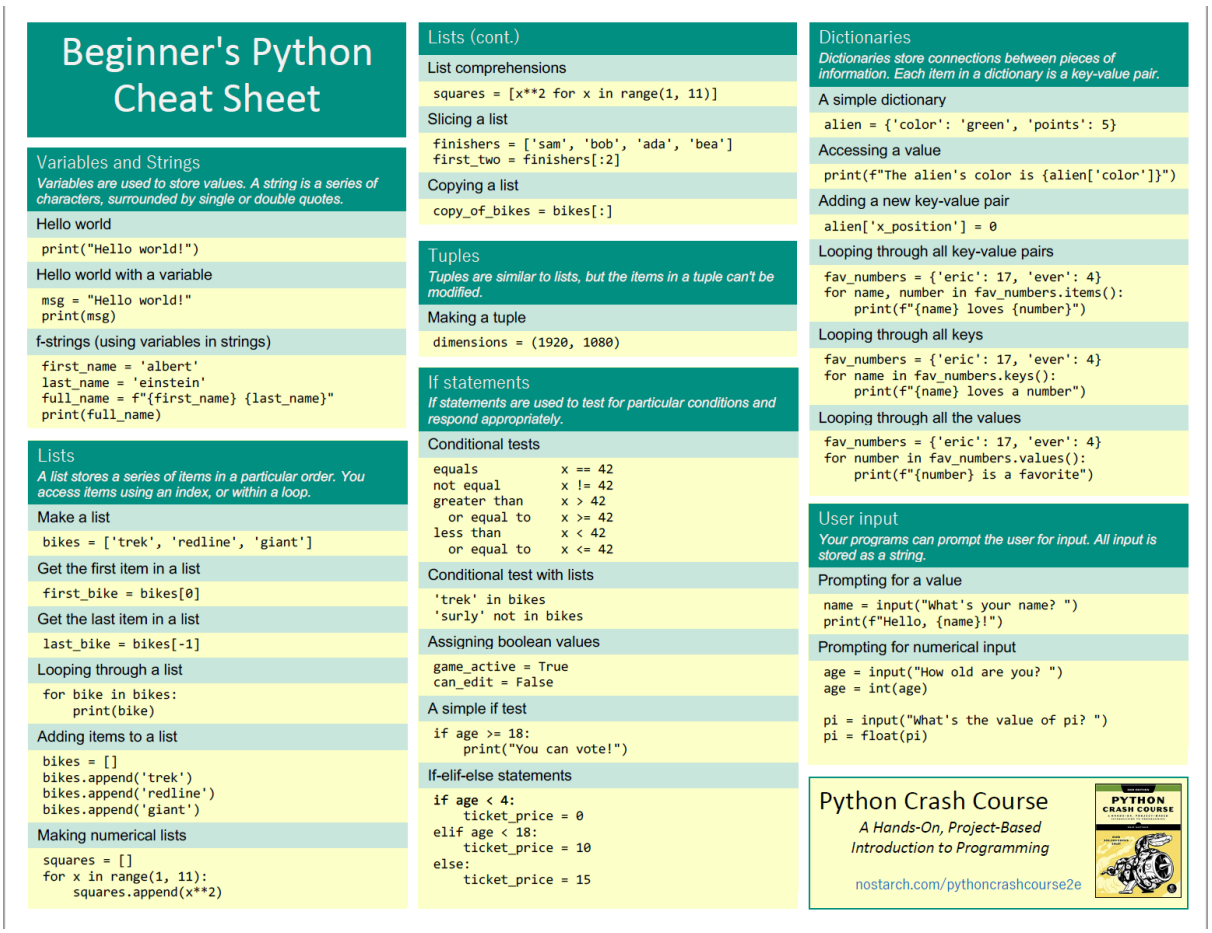
Reference/trouble-shooting websites

1. Google (**Allen**):
    - If you have a query about coding, or if you have an error message you can't comprehend, just Google it as your first-line solution.
2. Stack Overflow (**Laurence**):
    - *Description* - free, open-source code-checking, advice and troubleshooting.
    - *Why you recommend it* - if you have a coding problem and you ask Google, it is likely to have been answered in detail on Stack Overflow
    - *Cost* - a computer and an internet connection
3. W3Schools (**Edmond**):
    - *Description* - theory + exercises to nail the basics
    - *Why you recommend it* - clear, structured layout of concepts with good examples. Not just for Python either.
    - *Cost* - the ATP in your finger muscles to click to the website and do the exercises.
4. Application-specific (**Laurence**): any official website for a library e.g. scipy/numpy, even python.org
    - *Description* - these websites contain the documentation and use-cases for the modules and libraries you want to use, which range from first principles, to tutorial-like examples. If you want to tackle a problem with code, it is likely someone has already tried, and built code to scaffold your problems.

    - *Why you recommend it* - easy to ask Google about how to solve your problem using Python, then follow the links to the libraries and examples you can use
    - *Cost* - a computer and an internet connection
5. YouTube (**Daniel**):
    - *Description* - No more needs to be said
    - *Why you recommend it* - if you are a visual learner and prefer people to explain things to you verbally but can't be bothered investing yourself in online courses or walls of texts, YouTube can offer very helpful walkthroughs and examples that can help you with your specific queries
    - *Cost* - phone/computer and an internet connection
6. Reddit (**Daniel**):
    - *Description* - an all-in-one website with tons of discussions for just about any topic you can think of. Access through website or as an app on your phone
    - *Why you recommend it* - specific subreddits such as r/WebDev, r/AskProgramming, and r/LearnProgramming can be useful to be exposed to different unique and interesting things that other redditors might be involved in or have found out that can be helpful to you.
    - *Cost* - phone/computer and an internet connection, time as you might get distracted
7. Python Code Example (**Benny**)
    - *Description* - You can search for code directly, and you can type in code examples that you want to learn.
    - *Why you recommend it* - For instance, if you want to know how to read .csv file in Python, you just need to type in 'read csv' in the search box. Then, it gives you tons of examples, which are all extracted from open source projects.
    - *Cost* - phone/computer and an internet connection

Cheatsheets

1. Python Cheatsheet - Python Cheatsheet (**Rory**)
2. Cheat Sheets - Python Crash Course, 2nd Edition (**Rory**)



Beginner's Python Cheat Sheet

Variables and Strings

Variables are used to store values. A string is a series of characters, surrounded by single or double quotes.

Hello world

```
print("Hello world!")
```

Hello world with a variable

```
msg = "Hello world!"
print(msg)
```

f-strings (using variables in strings)

```
first_name = 'albert'
last_name = 'einstein'
full_name = f"{first_name} {last_name}"
print(full_name)
```

Lists

A list stores a series of items in a particular order. You access items using an index, or within a loop.

Make a list

```
bikes = ['trek', 'redline', 'giant']
```

Get the first item in a list

```
first_bike = bikes[0]
```

Get the last item in a list

```
last_bike = bikes[-1]
```

Looping through a list

```
for bike in bikes:
    print(bike)
```

Adding items to a list

```
bikes = []
bikes.append('trek')
bikes.append('redline')
bikes.append('giant')
```

Making numerical lists

```
squares = []
for x in range(1, 11):
    squares.append(x**2)
```

Lists (cont.)

List comprehensions

```
squares = [x**2 for x in range(1, 11)]
```

Slicing a list

```
finishers = ['sam', 'bob', 'ada', 'bea']
first_two = finishers[:2]
```

Copying a list

```
copy_of_bikes = bikes[:]
```

Tuples

Tuples are similar to lists, but the items in a tuple can't be modified.

Making a tuple

```
dimensions = (1920, 1080)
```

If statements

If statements are used to test for particular conditions and respond appropriately.

Conditional tests

```
equals            x == 42
not equal         x != 42
greater than      x > 42
  or equal to     x >= 42
less than         x < 42
  or equal to     x <= 42
```

Conditional test with lists

```
'trek' in bikes
'surly' not in bikes
```

Assigning boolean values

```
game_active = True
can_edit = False
```

A simple if test

```
if age >= 18:
    print("You can vote!")
```

If-elif-else statements

```
if age < 4:
    ticket_price = 0
elif age < 18:
    ticket_price = 10
else:
    ticket_price = 15
```

Dictionaries

Dictionaries store connections between pieces of information. Each item in a dictionary is a key-value pair.

A simple dictionary

```
alien = {'color': 'green', 'points': 5}
```

Accessing a value

```
print(f"The alien's color is {alien['color']}")
```

Adding a new key-value pair

```
alien['x_position'] = 0
```

Looping through all key-value pairs

```
fav_numbers = {'eric': 17, 'ever': 4}
for name, number in fav_numbers.items():
    print(f"{name} loves {number}")
```

Looping through all keys

```
fav_numbers = {'eric': 17, 'ever': 4}
for name in fav_numbers.keys():
    print(f"{name} loves a number")
```

Looping through all the values

```
fav_numbers = {'eric': 17, 'ever': 4}
for number in fav_numbers.values():
    print(f"{number} is a favorite")
```

User input

Your programs can prompt the user for input. All input is stored as a string.

Prompting for a value

```
name = input("What's your name? ")
print(f"Hello, {name}!")
```

Prompting for numerical input

```
age = input("How old are you? ")
age = int(age)

pi = input("What's the value of pi? ")
pi = float(pi)
```

Python Crash Course

A Hands-On, Project-Based Introduction to Programming

nostarch.com/pythoncrashcourse2e

3. Cheat sheets for data scientists (**Ryan**)
4. Cheat Sheets Archives – Dataquest (**Ryan**)

Bonus:

1. Screeps (**Daniel**):
    - *Description* - A real-time strategy game where you interact with the game via Javascript. Perfect if you are a fan of real time strategy games and you want to improve your Javascript skills.
    - *Cost* - A$21.50 on Steam