

plot_squared() - NumPy + Matplotlib Plotting

```
import numpy as np
import matplotlib.pyplot as plt

def plot_squared(n, x0, x1):
    xs = np.linspace(x0, x1, n)
    ys = xs * xs
    plt.plot(xs, ys)
    plt.title("y = x * x")
    plt.xlabel("x")
    plt.ylabel("y")
    plt.grid(True)
    plt.show()
```

afunction() - Nested While Loops

```
def afunction(value1, value2):
    op1 = 0
    while op1 < value1:
        op2 = value1
        while op2 > value2:
            result = op1 * op2
            print(f"{op1:3} x {op2:3} = {result:3}")
            op2 -= 1
        op1 += 1
```

File Max Finder

```
'ye'
def main():
    'ye'
    filename = input("Filename? ")
    with open(filename, "r") as file:
        max_int = None
        for line in file:
            line = int(line)
            if max_int == None or line > max_int:
                max_int = line
        print(f"Largest number found: {max_int}")
    main()
```

Files

```
def case_fixer(input_name,
output_name):
    "ye"
    with open(input_name, 'r') as
infile, open(output_name, 'w') as
outfile:
        for line in infile:
            cleaned_line = line.strip().lower()
            outfile.write(cleaned_line + '\n')

line.strip() # Remove leading/t-
railing whitespace
line.split() # Split into list of
words
line.split(',') # Split by comma
d.lower(), .upper(), .title() # Case
transforms
```

outputs

```
def trongo(x):
    print(len(x))
    s = set(x)
    y = {'fo', 'to', 'fun'}
    if len(s.intersection(y)) == 1:
        s = s.union(y)
    print(", ".join(sorted(list(s))))
```

```
# Example usage:
trongo(['fo', 'fi', 'hasan', 'fi',
'hasan'])
# Output:
# 5
# fi, fo, fun, hasan, to
```

Student Class 1

```
class Student:
    def __init__(self, student_id,
usercode, first_name, family-
_name):
        self.student_id = int(student_id)
        self.usercode = str(usercode)
        self.first_name = str(first_name)
        self.family_name = str(famil-
y_name)
        self.courses = set()

    def __str__(self):
        return f"{self.student_id} {self.f-"
```

Dicts

```
def inverted_word_counts(wor-
d_count_dict):
    "ye"
    inverted = {}
    for word, word_count in word_c-
ount_dict.items():
        if word_count in inverted:
            inverted[word_count].append-
(word)
        else:
            inverted[word_count] = [word]
    for words in inverted.values():
        words.sort()
    return inverted

def find_key(input_dict, value):
    "ye"
    for key, val in input_dict.items():
        if val == value:
            return key
    return None
```

Bar Graph

```
import numpy as np
import matplotlib.pyplot as plt

def main():
    rainfalls = np.loadtxt('laketaylor-
station2005.txt', delimiter=',',
skiprows=9, usecols=3)
    days = np.arange(1, len(rainf-
alls) + 1)
    axes = plt.axes()
    axes.bar(days, rainfalls)
    axes.set_title("Lake Taylor
Station Rainfalls, 2005")
    axes.set_xlabel("Day number")
    axes.set_ylabel("Rainfall (mm)")
    axes.grid(True)
    plt.show()

main()
```

Student Class 2

```
def enrol_course(self, course):
    self.courses.add(course)

def is_enrolled_in(self, course):
    return course in self.courses

def main():
    astudent = Student(1234567, "-
ffn127", "first NAMES", "Family
name")
    print(type(astudent))
    print(astudent)
    astudent.enrol_course("COSC1-
21")
    print(astudent.is_enrolled_in("-
COSC121"))
    print(astudent.is_enrolled_in("-
COSC122"))
    print(astudent.get_email_addre-
ss())

main()
```

dict

```
def frequencies(input_str):
    "ye"
    count_dict = {}
    for char in input_str:
        if char in count_dict:
            count_dict[char] += 1
        else:
            count_dict[char] = 1
    return count_dict
```

Classes

```
class Whatsit:
    def __init__(self, grade, name,
health):
        allowed = {'Sooglet', 'Throve',
'Plaguelet'}
        if grade not in allowed:
            raise ValueError('Invalid Whatsit
grade')
        self.grade = grade
        self.name = name
        self.health = max(float(health),
0)
```

```
amily_name.upper()), {self.first_ -  
name.capitalize()})"
```

```
def get_email_address(self):  
return f"{self.usercode}@ucli-  
ve.ac.nz"
```

```
def __str__(self):  
return f"{self.name} ({self.gr-  
ade}), health = {self.health:.1f}"
```

```
def damage(self, amount):  
self.health = max(self.health -  
amount, 0)
```



By **hasansheikh1**

cheatography.com/hasansheikh1/

Not published yet.

Last updated 18th June, 2025.

Page 1 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>