CS409
Software Testing

TAN, Shin Hwei
陈馨慧
Southern University of Science and Technology

Slides adapted from cs4218 in NUS

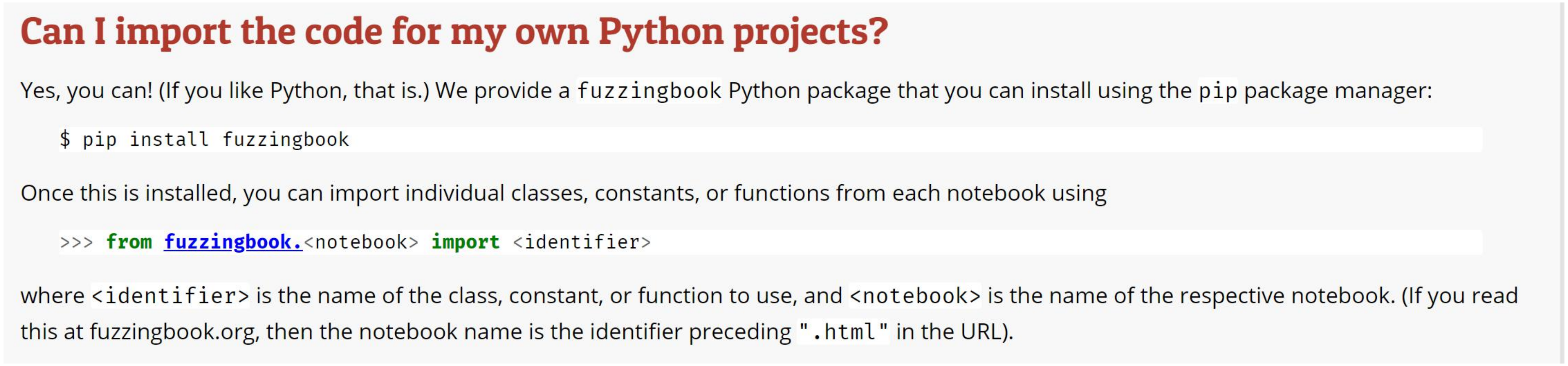
Fuzzing

Slides Adapted from
https://www.fuzzingbook.org/slides/Fuzzer.slides.html

Fuzzing Architecture

- Two important classes
  - Fuzzer as a base class for fuzzers; and
  - Runner as a base class for programs under test

1st Step(a): Run Fuzzingbook in your computer



Can I import the code for my own Python projects?

Yes, you can! (If you like Python, that is.) We provide a `fuzzingbook` Python package that you can install using the `pip` package manager:

```
$ pip install fuzzingbook
```

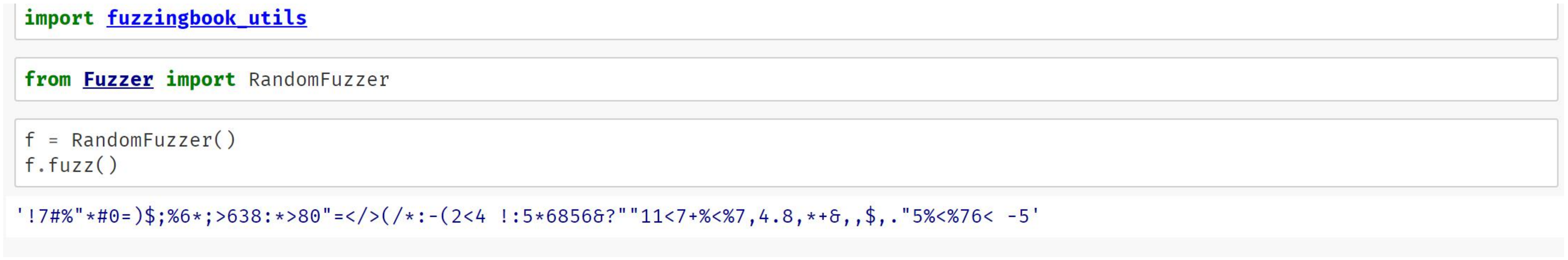
Once this is installed, you can import individual classes, constants, or functions from each notebook using

```
>>> from fuzzingbook.<notebook> import <identifier>
```

where `<identifier>` is the name of the class, constant, or function to use, and `<notebook>` is the name of the respective notebook. (If you read this at fuzzingbook.org, then the notebook name is the identifier preceding `".html"` in the URL).

1st Step (If you don’t want to install this in your computer): Run Fuzzingbook in notebook

Go to the following link:
https://notebooks.gesis.org/binder/jupyter/user/uds-se-fuzzingbook-xneda7rm/notebooks/docs/notebooks/Untitled.ipynb?kernel_name=python3



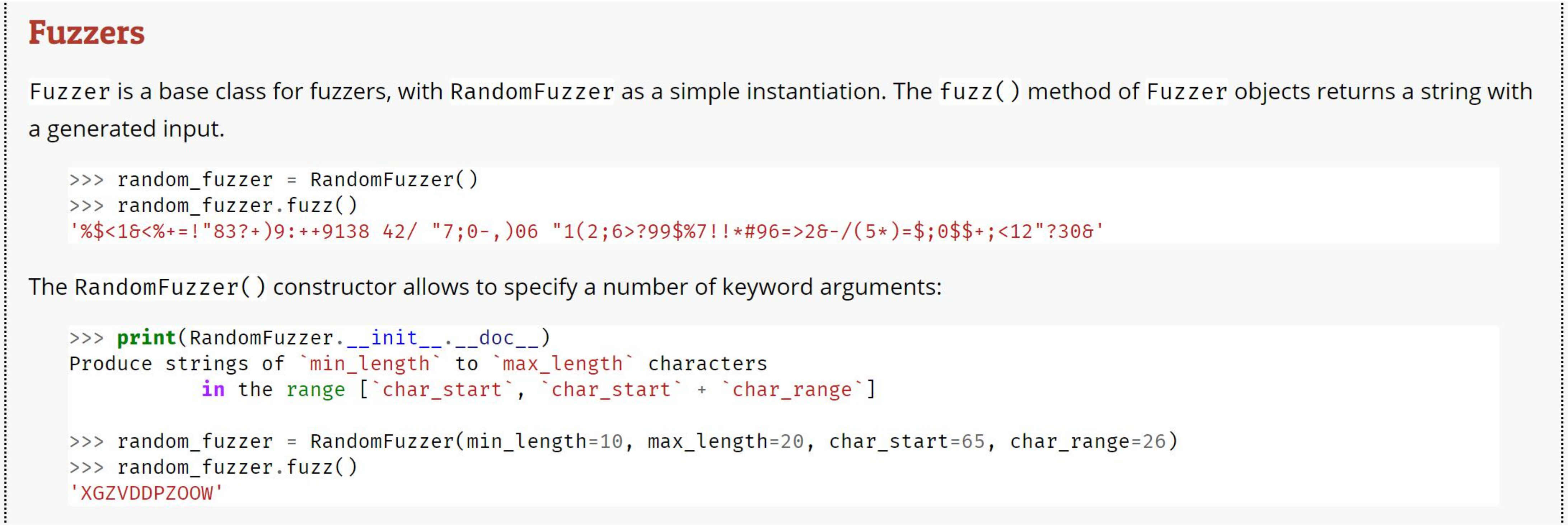
```
import fuzzingbook_utils
```

```
from Fuzzer import RandomFuzzer
```

```
f = RandomFuzzer()
f.fuzz()
```

```
'!7#%"*#0=)$;%6*;>638:*>80"=</>(/*:-(2<4 !:5*68568?""11<7+%<%7,4.8,**+&,,$,."5%<%76< -5'
```

Fuzzer class



Fuzzers

`Fuzzer` is a base class for fuzzers, with `RandomFuzzer` as a simple instantiation. The `fuzz()` method of `Fuzzer` objects returns a string with a generated input.

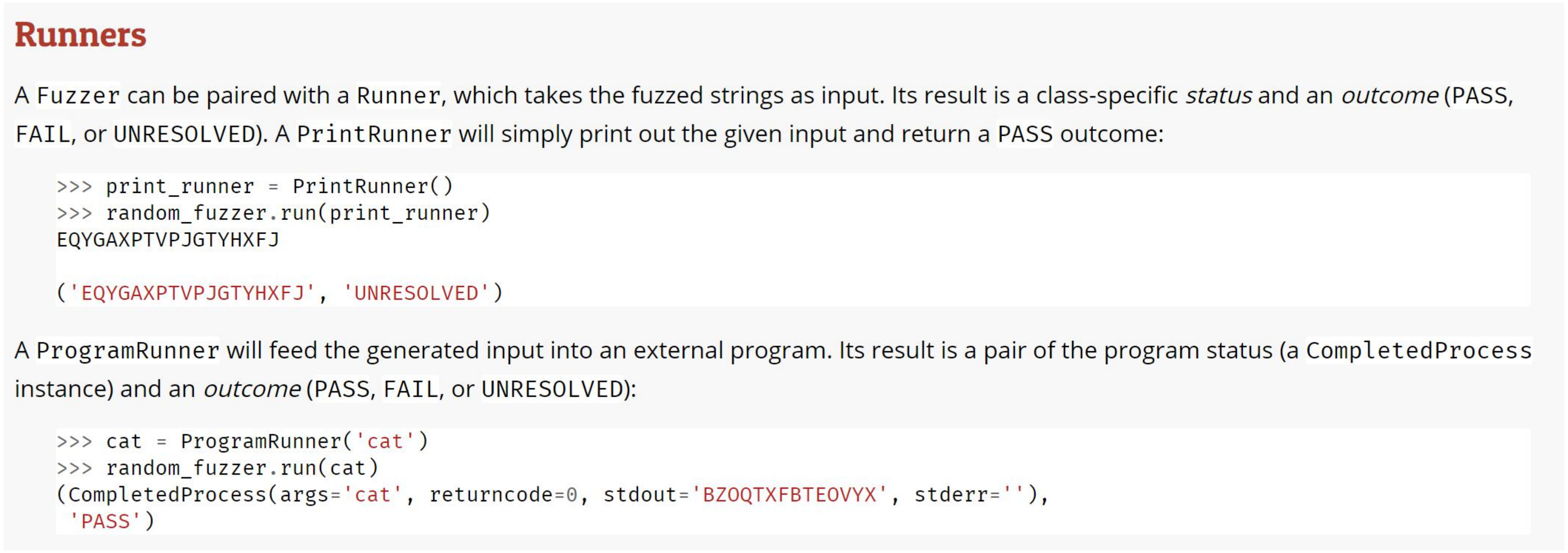
```
>>> random_fuzzer = RandomFuzzer()
>>> random_fuzzer.fuzz()
'%$<1&<%+=!"83?+)9:++9138 42/ "7;0-,)06 "1(2;6>?99$%7!!*#96=>2&-/(5*)=$;0$$+;<12"?30&'
```

The `RandomFuzzer()` constructor allows to specify a number of keyword arguments:

```
>>> print(RandomFuzzer.__init__.__doc__)
Produce strings of `min_length` to `max_length` characters
           in the range [`char_start`, `char_start` + `char_range`]

>>> random_fuzzer = RandomFuzzer(min_length=10, max_length=20, char_start=65, char_range=26)
>>> random_fuzzer.fuzz()
'XGZVDDPZOOW'
```

Runner class



Runners

A `Fuzzer` can be paired with a `Runner`, which takes the fuzzed strings as input. Its result is a class-specific *status* and an *outcome* (`PASS`, `FAIL`, or `UNRESOLVED`). A `PrintRunner` will simply print out the given input and return a `PASS` outcome:

```
>>> print_runner = PrintRunner()
>>> random_fuzzer.run(print_runner)
EQYGAXPTVPJGTYHXFJ

('EQYGAXPTVPJGTYHXFJ', 'UNRESOLVED')
```

A `ProgramRunner` will feed the generated input into an external program. Its result is a pair of the program status (a `CompletedProcess` instance) and an *outcome* (`PASS`, `FAIL`, or `UNRESOLVED`):

```
>>> cat = ProgramRunner('cat')
>>> random_fuzzer.run(cat)
(CompletedProcess(args='cat', returncode=0, stdout='BZOQTXFBTEOVYX', stderr=''),
 'PASS')
```

Let’s us build a fuzzer!

- Idea: Produce random characters, adding them to a buffer string variable (out), and finally returning the string.
- This implementation uses the following Python features and functions:
  - random.randrange(start, end) – return a random number [start, end)
  - range(start, end) – create a list with integers in the range [start, end). Typically used in iterations.
  - for elem in list: body – execute body in a loop with elem taking each value from list.
  - for i in range(start, end): body – execute body in a loop with i from start to end − 1.
  - chr(n) – return a character with ASCII code n

Simple Fuzzer

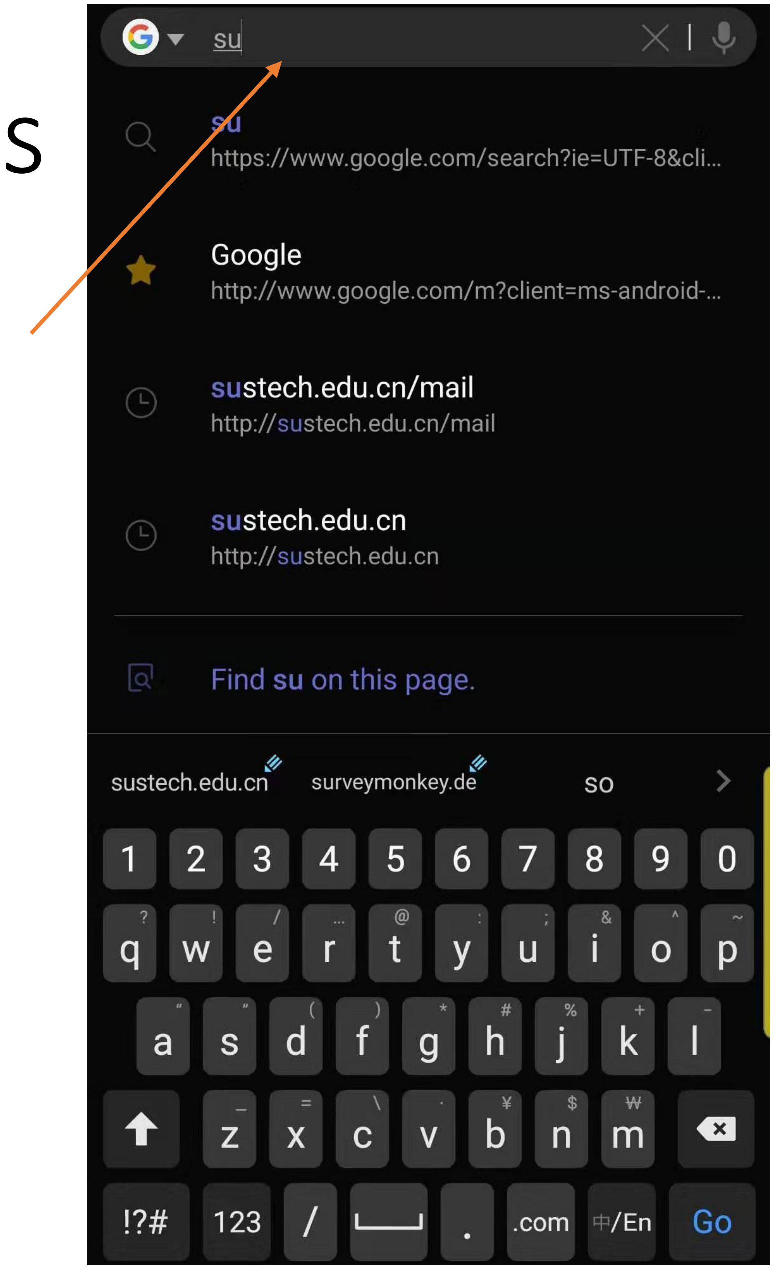
- Generate random length (up to max length) of random string

```
def fuzzer(max_length=100, char_start=32, char_range=32):
    """A string of up to `max_length` characters
      in the range [`char_start`, `char_start` + `char_range`]"""
    string_length = random.randrange(0, max_length + 1)
    out = ""
    for i in range(0, string_length):
        out += chr(random.randrange(char_start, char_start + char_range))
    return out
```

- How to call this fuzzer?
  - Ex. If we want to produce a series of lowercase letters. We use ord(c) to return the ASCII code of the character c.
    - fuzzer(1000, ord('a'), 26)

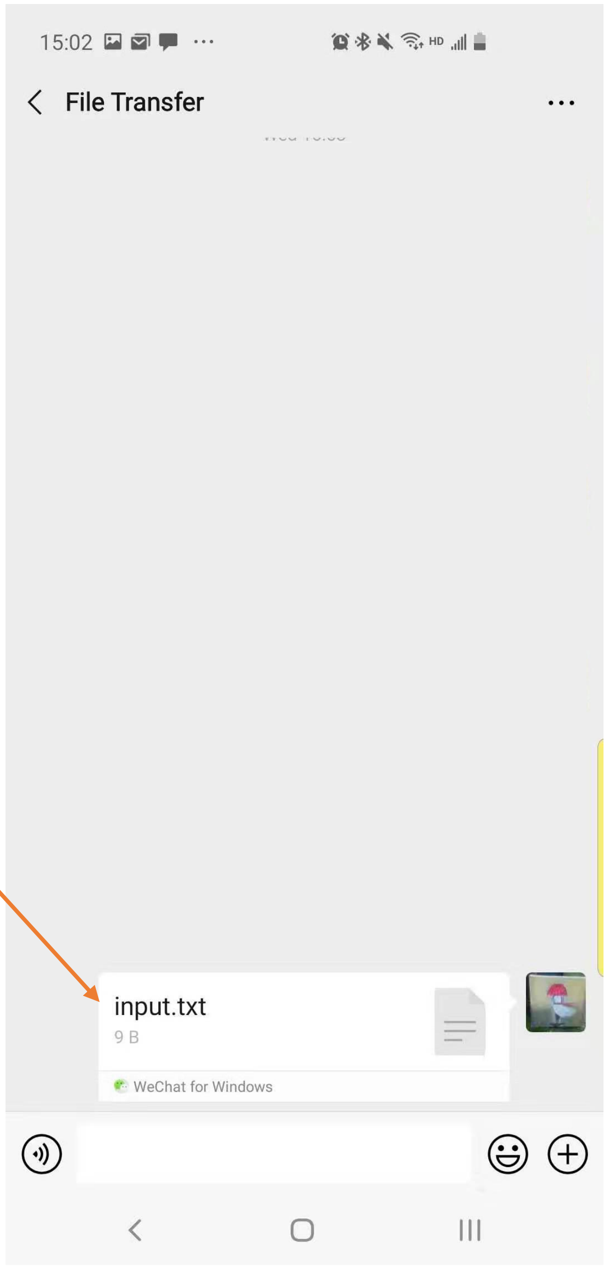
Apply this to your selected apps

- Identify the text box in your apps



Apply this to your selected apps

- Identify the input file in your apps


15:02

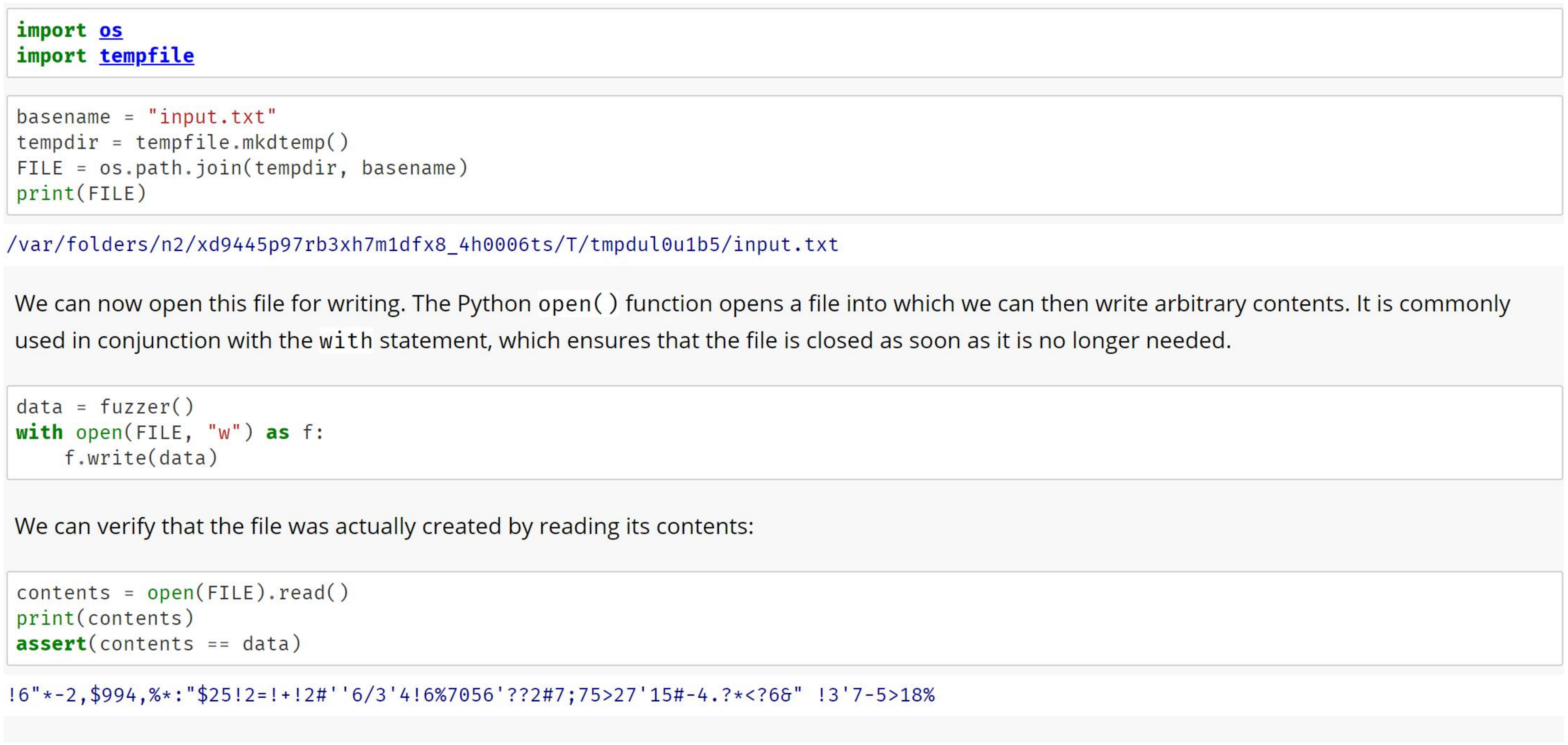
File Transfer

input.txt

9 B

WeChat for Windows

Creating input files



```python
import os
import tempfile
```

```python
basename = "input.txt"
tempdir = tempfile.mkdtemp()
FILE = os.path.join(tempdir, basename)
print(FILE)
```

```
/var/folders/n2/xd9445p97rb3xh7m1dfx8_4h0006ts/T/tmpdul0u1b5/input.txt
```

We can now open this file for writing. The Python `open()` function opens a file into which we can then write arbitrary contents. It is commonly used in conjunction with the `with` statement, which ensures that the file is closed as soon as it is no longer needed.

```python
data = fuzzer()
with open(FILE, "w") as f:
    f.write(data)
```

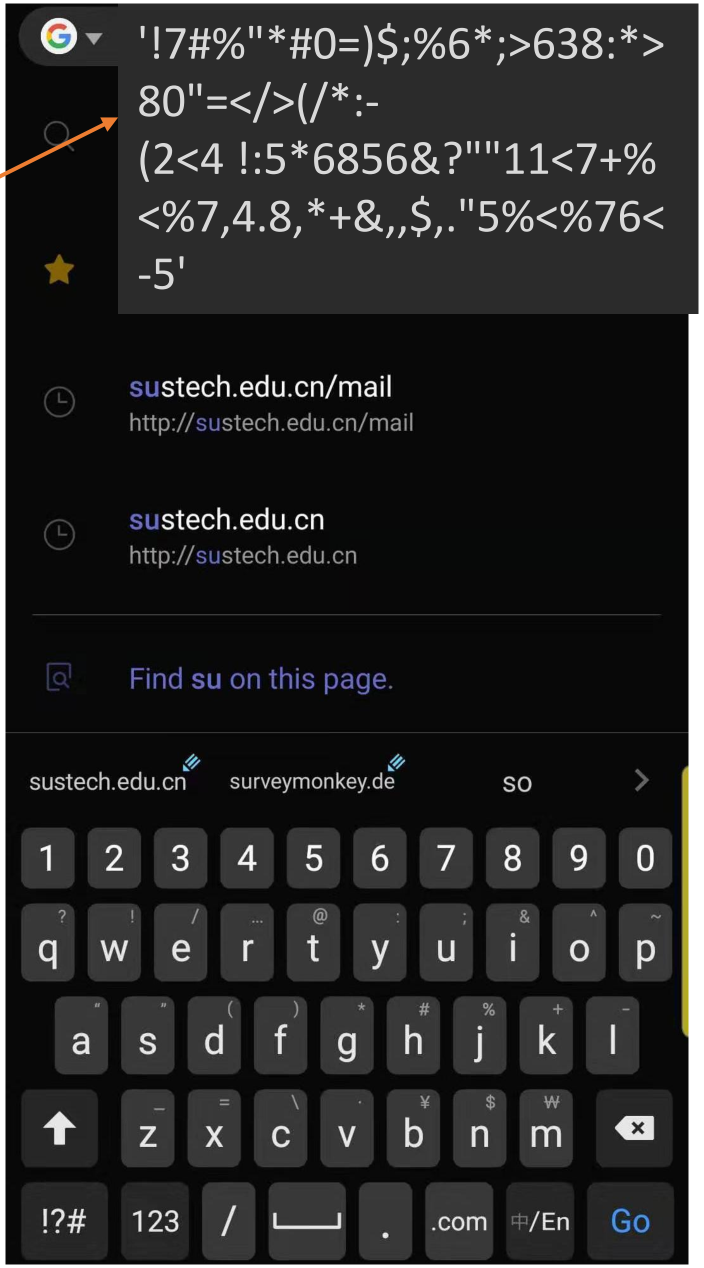
We can verify that the file was actually created by reading its contents:

```python
contents = open(FILE).read()
print(contents)
assert(contents == data)
```

```
!6"*-2,$994,%*:"$25!2=!+!2#''6/3'4!6%7056'??2#7;75>27'15#-4.?*<?6&" !3'7-5>18%
```

Get fuzzed Inputs

- Give the fuzzed inputs to your app



```
'!7#%"*#0=)$;%6*;>638:*>
80"=</>(/*:-
(2<4 !:5*6856&?""11<7+%
<%7,4.8,*+&,,$,."5%<%76<
-5'
```

sustech.edu.cn/mail
http://sustech.edu.cn/mail

sustech.edu.cn
http://sustech.edu.cn

Find su on this page.

Could we do better than random inputs?

- **Fuzzing with Grammars**
- Adapted from https://www.fuzzingbook.org/html/Grammars.html

Defining grammar: Example

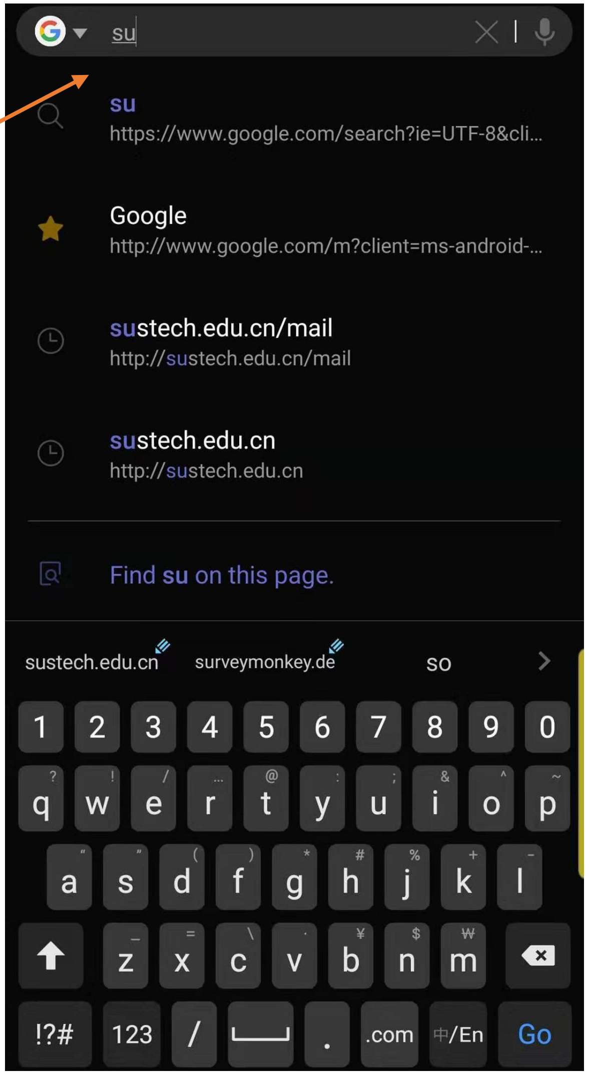
```
import fuzzingbook_utils
from Grammars import simple_grammar_fuzzer
US_PHONE_GRAMMAR = {
    "<start>": ["<phone-number>"],
    "<phone-number>": ["(<area>)<exchange>-<line>"],
    "<area>": ["<lead-digit><digit><digit>"],
    "<exchange>": ["<lead-digit><digit><digit>"],
    "<line>": ["<digit><digit><digit><digit>"],
    "<lead-digit>": ["2", "3", "4", "5", "6", "7", "8", "9"],
    "<digit>": ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9"]
 }
[simple_grammar_fuzzer(US_PHONE_GRAMMAR) for i in range(5)]
```

Grammar for US phone number

- https://www.fuzzingbook.org/html/Grammars.html

Write your own grammar for your input

- If your app have simple text input, then write grammar for the simple text input
- If you have no simple text input (e.g., need to write grammar for a photo), then write grammar for the web address



su

su
https://www.google.com/search?ie=UTF-8&cli...

Google
http://www.google.com/m?client=ms-android-...

sustech.edu.cn/mail
http://sustech.edu.cn/mail

sustech.edu.cn
http://sustech.edu.cn

Find su on this page.

sustech.edu.cn surveymonkey.de so

Get fuzzed Inputs from the grammar fuzzer

- Get 5 valid inputs from the grammar fuzzer

```
import fuzzingbook_utils
from Grammars import simple_grammar_fuzzer
MY_GRAMMAR = {
    …
 }
[simple_grammar_fuzzer(MY_GRAMMAR) for i in range(5)]
```

- Give the fuzzed inputs to your app

Submit your fuzzed inputs

- https://classroom.github.com/a/WOPbCjnZ
- Add a README.md and answer the following questions your app
  - What is the text input that you try to fuzz?
    - If there is no text input for your app, use the Google Chrome App or Baidu app to answer the question for this lab
  - Add a screenshot for showing the text input?
  - What are the random fuzzed inputs from random fuzzer?
  - What are the fuzzed inputs from grammar fuzzer?
  - Report if you find any new crash
    - Use delta debugging to minimize the fuzzed inputs

    - **Don't forget to write your student id and name in the README.md!**