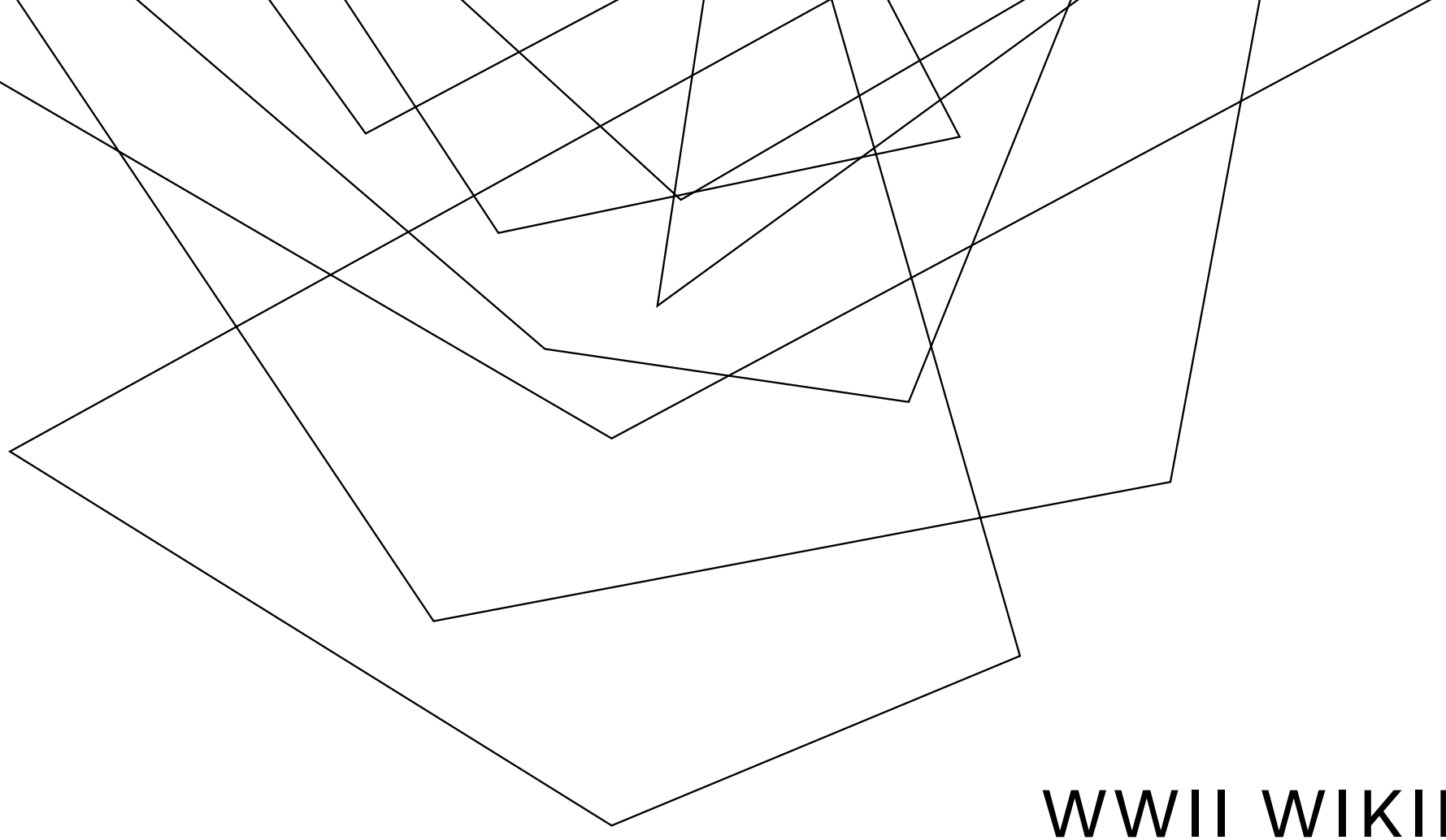

WWII WIKIPEDIA SCRAPER

Berjer Ding
Toronto, Canada

PREMISE

- Scrapes information from Wikipedia
- Written in Python
- Centered around World War 2 pages as boundary
- Testable

project.py test_project.py

```python
import wikipediaapi, re, random, sys


wiki = wikipediaapi.Wikipedia('en')


def main():
    x = input("Would you like to know about a |specific| event, a |random| event, or an event in a |general| year? ")
    if x.lower() == "specific":
        print(specific(input("What would you like to know about? ")))
    elif x.lower() == "random":
        randomlist = []
        battles = wiki.page("List of World War II battles").links
        for link in battles.keys():
            randomlist.append(link)
        i = random.randint(0,len(randomlist))
        print(random_event(i))
    elif x.lower() == "general":
        print(general_event())
    else:
        print("Please enter |specific| or |random| or |general| to continue")


def specific_event(i):
    page = wiki.page(i)
    if page.exists():
        if matches := re.search(r"1939|194[0-5]", page.summary):
            return page.summary
        else:
            return "Page is not about the Second World War or WW2-adjacent."
    else:
        return "Page does not exist"


def random_event(x):
    randomlist = []
    battles = wiki.page("List of World War II battles").links
    for link in battles.keys():
        randomlist.append(link)
    battle_page = wiki.page(randomlist[x])
    return battle_page.summary


def general_event():
    generallist = []
    portal = wiki.page("Category:World War II").categorymembers
    for category in portal.keys():
        generallist.append(category.replace("Category:",""))
    print(generallist)
    while True:
        inputted = input("")
        if inputted in generallist:
            generallist = []
            portal = wiki.page(f"Category:{inputted}").categorymembers
            for category in portal.keys():
                generallist.append(category.replace("Category:",""))
            if generallist == []:
                return wiki.page(inputted).summary
            else:
                print(generallist)
        else:
            sys.exit("Incorrect selection")


if __name__ == "__main__":
    main()
```

PRELIMINARIES

```
1 import wikipediaapi, re, random, sys
2
3
4 wiki = wikipediaapi.Wikipedia('en')
```

Importing 4 libraries:

1. Wikipedia API
2. Regular Expressions
3. Random Module
4. Sys Module

Shorten the Wikipedia API to the wiki keyword and in English

```
7  def main():
8      x = input("Would you like to know about a |specific| event, a |random| event, or an event in a |general| year? ")
9      if x.lower() == "specific":
10         print(specific(input("What would you like to know about? ")))S
11     elif x.lower() == "random":
12         randomlist = []
13         battles = wiki.page("List of World War II battles").links
14         for link in battles.keys():
15             randomlist.append(link)
16         i = random.randint(0,len(randomlist))
17         print(random_event(i))
18     elif x.lower() == "general":
19         print(general_event())
20     else:
21         print("Please enter |specific| or |random| or |general| to continue")
22
```

MAIN() FUNCTION

- For the user to get to the individual functions
- Three options: specific, random, and general
- Hardcoded to input the options

```python
def specific_event(i):
    page = wiki.page(i)
    if page.exists():
        if matches := re.search(r"1939|194[0-5]", page.summary):
            return page.summary
        else:
            return "Page is not about the Second World War or WW2-adjacent."
    else:
        return "Page does not exist"

```

SPECIFIC_EVENT(I)

- Easiest to make and test
- Receives as input “i” which is a topic the user inputted in main()
- Wikipedia API returns the page if it exists
- Re.search confirms if it is about WW2
- Returns summary of the page

RANDOM_EVENT(X)

- Takes as input a randomly generated number from main()
- Selects from the wiki page of WWII battles
- Uses the wiki page links as a list
- Returns a randomly selected page summary

```
35 def random_event(x):
36     randomlist = []
37     battles = wiki.page("List of World War II battles").links
38     for link in battles.keys():
39         randomlist.append(link)
40     battle_page = wiki.page(randomlist[x])
41     return battle_page.summary
42
```

```
43
44 def general_event():
45     generallist = []
46     portal = wiki.page("Category:World War II").categorymembers
47     for category in portal.keys():
48        generallist.append(category.replace("Category:",""))
49     print(generallist)A
50     while True:
51         inputted = input("")
52         if inputted in generallist:
53             generallist = []
54             portal = wiki.page(f"Category:{inputted}").categorymembers
55             for category in portal.keys():
56                 generallist.append(category.replace("Category:",""))
57             if generallist == []:
58                 return wiki.page(inputted).summary
59             else:
60                 print(generallist)
61         else:
62             sys.exit("Incorrect selection")
63
```

GENERAL_EVENT()

- Selects from the WWII category portal
- First captures the available categories on the page to output
- In a while-loop, allows user input to delve deeper into the categories
- Ends when a page is found, which is then outputted

```python
import pytest, project, mock, builtins

def test_specific_event():
    assert project.specific_event("List of French divisions in World War II") == "This is a listing of French divisions that served between 1939 and 1945."
    assert project.specific_event("Cat") == "Page is not about the Second World War or WW2-adjacent."
    assert project.specific_event("BerjerDing") == "Page does not exist"

def test_random_event():
    assert project.random_event(5) == "The Action at Bir el Gubi  (November 1941) (Arabic: بئر الغبي, romanized: Bir al-Ġubbiyy, lit. 'Well of the Depressed Terrain')


def test_general_event():
    with mock.patch.object(builtins, 'input', lambda _: 'USCGC Northland (WPG-49)'):
        assert project.general_event() == 'USCGC Northland (WPG-49) was a United States Coast Guard cruising class of gunboat especially designed for Arctic operations
    with mock.patch.object(builtins, 'input', lambda _: 'Cat'):
        with pytest.raises(SystemExit):
            project.general_event()
    with mock.patch.object(builtins, 'input', lambda _: 'BerjerDing'):
        with pytest.raises(SystemExit):
            project.general_event()
```

TESTING

- Testing specific_event() was confirming the inputted page matched the output summary
- Testing random_event() was through the parameter, which matched the argument page to the out page
- Testing general_event() was harder:
    - Required mocking the input of the user
    - Asserting that the mocked input returned the correct output