1. To add the current date to the text file `today.txt` as a string, we can use the following code:

```python

from datetime import date

with open('today.txt', 'a') as file:

file.write(str(date.today()))

```

This will open the file `today.txt` in append mode, and then write the current date as a string to the end of the file.

2. To read the text file `today.txt` into the string `today\_string`, we can use the following code:

```python

with open('today.txt', 'r') as file:

today\_string = file.read()

```

This will open the file `today.txt` in read mode, and then read the contents of the file into the string `today\_string`.

3. To parse the date from `today\_string`, we can use the `datetime.datetime.strptime()` method. Assuming that the date is in the format "YYYY-MM-DD", we can use the following code:

```python

from datetime import datetime

parsed\_date = datetime.strptime(today\_string.strip(), '%Y-%m-%d').date()

```

This will first remove any leading or trailing whitespace from the string `today\_string` using the `strip()` method, and then parse the date using the `strptime()` method with the format string `'%Y-%m-%d'`. The resulting `datetime.datetime` object is then converted to a `datetime.date` object using the `date()` method.

4. To list the files in the current directory, we can use the `os.listdir()` function. This function returns a list of all the files in the directory. We can use the following code:

```python

import os

files = os.listdir('.')

print(files)

```

This will list all the files in the current directory.

5. To create a list of all the files in the parent directory, we can use the `os.listdir()` function again, but this time with the parent directory as the argument. We can use the following code:

```python

import os

parent\_dir = os.path.abspath('..')

files = os.listdir(parent\_dir)

print(files)

```

This will first get the absolute path of the parent directory using the `os.path.abspath()` function, and then list all the files in that directory using the `os.listdir()` function.

6. To use multiprocessing to create three separate processes that wait a random number of seconds between one and five, print the current time, and then exit, we can use the `multiprocessing` module along with the `time` module. We can use the following code:

```python

import multiprocessing

import time

import random

from datetime import datetime

def wait\_random():

wait\_time = random.randint(1, 5)

time.sleep(wait\_time)

print(f"Process {multiprocessing.current\_process().name} waited for {wait\_time} seconds and finished at {datetime.now()}")

if \_\_name\_\_ == '\_\_main\_\_':

processes = []

for i in range(3):

p = multiprocessing.Process(target=wait\_random)

processes.append(p)

p.start()

for p in processes:

p.join()

```

This will define a function `wait\_random()` that waits for a random number of seconds between one and five using the `time.sleep()` function, prints the current time using the `datetime.now()` function, and then exits. We then create three separate processes using the `multiprocessing.Process()` class and start them using the `start()` method. Finally, we wait for all the processes to finish using the `join()` method.

7. To create a date object of our day of birth, we can use the `datetime.date()` constructor. Assuming that our date of birth