# Think Python

Exercise 16.1.
Write a function called mul_time that takes a Time object and a number and returns a new Time object that contains the product of the original Time and the number.

Then use mul_time to write a function that takes a Time object that represents the finishing time in a race, and a number that represents the distance, and returns a Time object that represents the average pace (time per mile).

---

```
def main():
        time = Time(1, 0, 0)
        print(time)
        print(average_pace(time, 2))

class Time:
        def __init__(self, hours=0, minutes=0, seconds=0.0):
                # Initialize the Time object with hours, minutes, and seconds
                self.hours = hours
                self.minutes = minutes
                self.seconds = seconds
                self.normalize_time()  # Normalize time components
                assert self.valid_time(), "Time object is not valid"  # Ensure validity

        def normalize_time(self):
                # Normalize seconds to minutes and minutes to hours
                if self.seconds >= 60:
                        self.minutes += int(self.seconds // 60)
                        self.seconds = self.seconds % 60
                if self.minutes >= 60:
                        self.hours += self.minutes // 60
                        self.minutes = self.minutes % 60

        def valid_time(self):
                # Check if the Time object has valid hour, minute, and second values
                if self.hours < 0 or self.minutes < 0 or self.seconds < 0:
                        return False
                if self.minutes >= 60 or self.seconds >= 60:
                        return False
                return True
```

```
        def __str__(self):
                # Return a string representation of the Time object
                return f"{self.hours}h {self.minutes}m {self.seconds:.1f}s"


def time_to_int(time_obj):
        # Convert a Time object to total seconds
        return int(time_obj.hours * 3600 + time_obj.minutes * 60 + time_obj.seconds)

def int_to_time(total_seconds):
        # Convert total seconds back to a Time object
        hours = total_seconds // 3600
        total_seconds %= 3600
        minutes = total_seconds // 60
        seconds = total_seconds % 60
        return Time(hours, minutes, seconds)

def mul_time(time_obj, multiplier):
        # Multiply a Time object by a number
        total_seconds = time_to_int(time_obj) * multiplier
        return int_to_time(total_seconds)

def average_pace(finishing_time, distance):
        # Calculate the average pace from finishing time and distance
        total_seconds = time_to_int(finishing_time)
        pace_seconds_per_mile = total_seconds / distance
        return int_to_time(pace_seconds_per_mile)

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

## Exercise 16.1 Output.

```
1h 0m 0.0s

0.0h 30.0m 0.0s
```

## Exercise 16.2.

The datetime module provides time objects that are similar to the Time objects in this chapter, but they provide a rich set of methods and operators. Read the documentation at http://docs.python.org/3/library/datetime.html .

1. Use the datetime module to write a program that gets the current date and prints the day of the week.

2. Write a program that takes a birthday as input and prints the user's age and the number of days, hours, minutes and seconds until their next birthday.

3. For two people born on different days, there is a day when one is twice as old as the other. That's their Double Day. Write a program that takes two birth dates and computes their Double Day.

4. For a little more challenge, write the more general version that computes the day when one person is n times older than the other.

---

```
from datetime import datetime
from datetime import timedelta

def main():
        print(current_day())
        print(next_birthday())

def current_day():
        # Gets current day
        today = datetime.today()
        return f"Today is a {today.strftime('%A')}"

def next_birthday():
        #       Calculates the date of the next birthday from a given birth date

        birthday_str = "14/02/1985"
        birthday = datetime.strptime(birthday_str, '%d/%m/%Y')

        today = datetime.today()
        next_birthday = birthday.replace(year=today.year)

        if next_birthday < today:
                next_birthday = next_birthday.replace(year=today.year + 1)

        time_to_next_birthday = next_birthday - today
```

```
        days = time_to_next_birthday.days
        seconds = time_to_next_birthday.seconds
        hours = seconds // 3600
        minutes = (seconds % 3600) // 60
        seconds = seconds % 60

        return (
                f"Original Birthday: {birthday}\n"
                f"Next Birthday: {next_birthday}\n"
                f"Time to next birthday: {days} days, {hours} hours, "
                f"{minutes} minutes, {seconds} seconds"
        )

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

## Exercise 16.2 Output.

```
Today is a Tuesday

Original Birthday: 1985-02-14 00:00:00

Next Birthday: 2025-02-14 00:00:00

Time to next birthday: 37 days, 14 hours, 23 minutes, 15 seconds
```