VARIABLES

- Pointers to Python objects with =,
- Can operate on them,
- Can create new ones from old ones.

IF-STATEMENTS

- Conditionally execute portions of code once,
- Checks a Boolean and then executes a code block.

WHILE-LOOPS

- Conditionally execute portions code of repeatedly while condition is true,
- Repeatedly checks a Boolean and then executes a code block.

FUNCTIONS

- Executable portion of code that can be used on demand,
- Can be passed variables,
- Indented portion of code is executed and can output Python objects with `return`,
- Defined with `def`.

```
import random


def simulate_queue(
        arrival_rate,
        service_rate,
        number_of_staff,
        time_period, limit):
    """
    Simulates one run of a queue and returns the proportion of customers
    waiting over a given limit. The parameters are:

    + arrival_rate
    + service_rate
    + number_of_staff
    + time_period
    + limit
    """
    number_of_customers = 0
    number_over_limit = 0
    now = 0
    server_available_dates = [0] * number_of_staff
    service_times = []

    while now < time_period:
        inter_arrival_time = random.expovariate(arrival_rate)
        now += inter_arrival_time
        number_of_customers += 1

        service_start_date = max(now, min(server_available_dates))

        service_time = random.expovariate(service_rate)
        service_end_date = service_start_date + service_time

        server_available_dates.append(service_end_date)
        server_available_dates.sort()
        server_available_dates = server_available_dates[-number_of_staff:]

        wait = service_start_date - now
        if wait > limit:
            number_over_limit += 1

    return number_over_limit / number_of_customers



def get_proportion_waiting_over_limit(
    arrival_rate=1.5,
    service_rate=0.15,
    number_of_staff=10,
    limit=0.5,
    time_period=31*24,
    number_of_repetitions=100):
    """
    Gives the average proportion of customers waiting over a given limit,
    over number_of_repetitions repetitions. The parameters are:

    + arrival_rate
    + service_rate
    + number_of_staff
    + time_period
    + limit
    + number_of_repetitions
    """
    proportions = []
    for repetition in range(number_of_repetitions):
        proportions.append(
            simulate_queue(
                arrival_rate=arrival_rate,
                service_rate=service_rate,
                number_of_staff=number_of_staff,
                limit=limit,
                time_period=time_period))
    return sum(proportions) / len(proportions)



get_proportion_waiting_over_limit()
```

BOOLEANS

- A type of variable,
- `True` or `False`,
- Can operate on them (`==`, `and`, `or`, `!=`, ...),
- Can be created from other variables with operators (==, !=, <, >, <=, ...).

LISTS

- Ordered collections of pointers (variables),
- They have methods: `sort`, `min`, `len`, `max` ...,
- Indexable.

FOR-LOOPS

- Repeatedly execute portions of code for every element in a collection,
- Current element is usable as a variable,
- Repeatedly executes indented code block.

OTHER?

Imports

Comments

Docstrings

Default arguments