What have we learned so far?

Control Structures

- Sequences of statements

```
fst_sentence = 'Call me Ishmael.'
fst_sentence = fst_sentence.split()
fst_sentence[-1] = 'Jens.'
print(' '.join(fst_sentence))
```

- Conditional statements

```
if filename == 'ropf.zip':
    do_something()
else:
    do_something_else()
```

- Grouping of repetitive tasks in loops

```
for element in collection:
    do_something_with(element)

while condition:
    do_something()
```

More on loops in a moment...

Datastructures

- Basic Datatypes: Integer, Float, String

```
print(42)
print(42.0)
print('Fourtytwo!')
```

- Lists

```
print([42, 42.0, 'Fourtytwo!'])
```

More on loops

Loops are a way to **repeat the same code** again and again, but for different inputs.

Imagine that you run a chatroom, and you want to print how many users are online *every time* someone logs in.

That message would be `'Users online: X'` where `X` is the number of users. In Python that would be

```
print('Users online: ' + str(users_online))
```

... where the **variable** `users_online` is an integer

So in *pseudocode* we need something like:

```
set users_online to 0
Everytime a user logs in:
   print 'Users online X'
   wait for another user to log in
```

This code has a bug. Fix it and then run this through the debugger step by step:

```
online_users = 0

while online_users < 100:
    message = 'Users online: ' + users_online
    print(message)
    online_users += 1
```

Run this through the debugger step by step:

```
from turtle import forward, right


while True:
    forward(100)
    right(45)
```

In [2]:

```
my_list = [1, 2, 3, 4]
for element in my_list:
    print(element)
```

```
1
2
3
4
```

Run this through the debugger step by step:

```
my_list = [1, 2, 3, 4]
for i in my_list:
    print(i)
```

Run this through the debugger step by step:

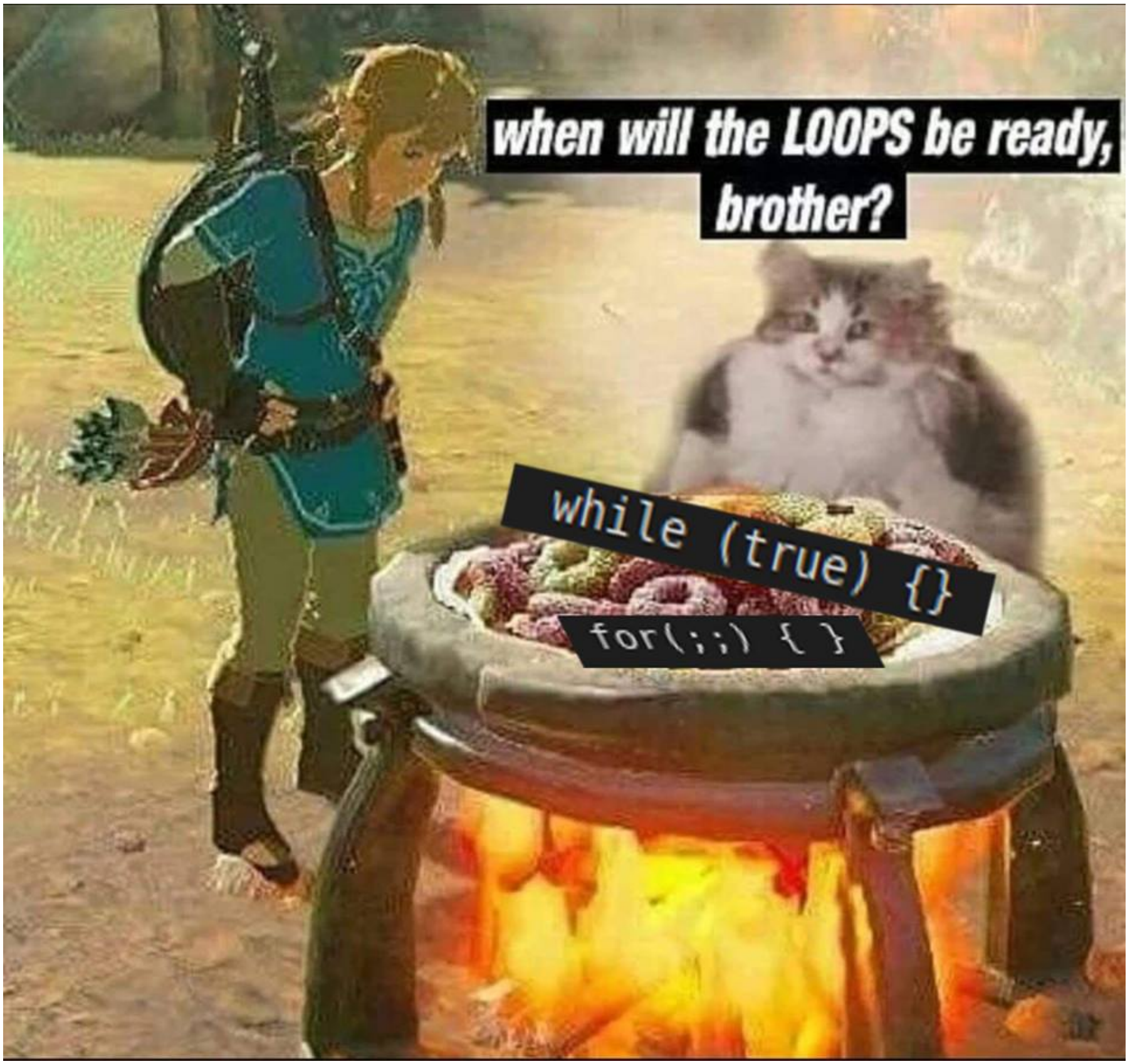
```
from turtle import forward, left


turtle_moves = [100, 200, 100, -20]
for move in turtle_moves:
    forward(move)
    left(90)
```

The infinite loop

```
while True:
    ...
```

Seen on SWU memes



General Feedback

What should be the name of your hand-in in case your ITU login is `abcd`?

- `abcd@itu.dk.zip (mailto:%60abcd@itu.dk.zip)`
- `group_with_funny_name.zip`
- `Wiglaf.txt`
- `abcd.zip`

Group Names

There are humans reading your assignments. Choose your group names so that you would not have a problem saying it when having coffee and cake with your grandparents on a Sunday afternoon.

Hand-in format.

The ZIP file contains:

```
├── A.py
├── B.py
├── C.py
└── D.py
```

And not:

```
├── A.zip
├── B.py
├── C.py
└── Another.zip
```

I found some code online...

Great! Take it, read it, understand it, modify it according to your use case and use the modified code.

Do not just copy and paste code without being completely sure what it does.

Pseudocode is almost code

In case you are in doubt, write pseudocode instead of code. It helps you organize and express your thoughts and it might allow the TA's to give more valuable feedback as they might understand better what you are trying to do.

In the same style, it is a good idea to write in comments where exactly you are lost and what you would like to do there.

What have we learned so far?

Control Structures

- Sequences of statements

```
fst_sentence = 'Call me Ishmael.'
fst_sentence = fst_sentence.split()
fst_sentence[-1] = 'Jens.'
print(' '.join(fst_sentence))
```

- Conditional statements

```
if filename == 'ropf.zip':
    do_something()
else:
    do_something_else()
```

- Grouping of repetitive tasks in loops

```
for element in collection:
    do_something_with(element)

while condition:
    do_something()
```

More on loops in a moment...

Datastructures

- Basic Datatypes: Integer, Float, String

```
print(42)
print(42.0)
print('Fourtytwo!')
```

- Lists

```
print([42, 42.0, 'Fourtytwo!'])
```

More on loops

Loops are a way to **repeat the same code** again and again, but for different inputs.

Imagine that you run a chatroom, and you want to print how many users are online *every time* someone logs in.

That message would be `'Users online: X'` where `X` is the number of users. In Python that would be

```
print('Users online: ' + str(users_online))
```

... where the **variable** `users_online` is an integer

So in *pseudocode* we need something like:

```
set users_online to 0
Everytime a user logs in:
   print 'Users online X'
   wait for another user to log in
```

This code has a bug. Fix it and then run this through the debugger step by step:

```
online_users = 0

while online_users < 100:
    message = 'Users online: ' + users_online
    print(message)
    online_users += 1
```

Run this through the debugger step by step:

```
from turtle import forward, right


while True:
    forward(100)
    right(45)
```

In [2]:

```
1
2
3
4
```

Run this through the debugger step by step:

```
my_list = [1, 2, 3, 4]
for i in my_list:
    print(i)
```

Run this through the debugger step by step:

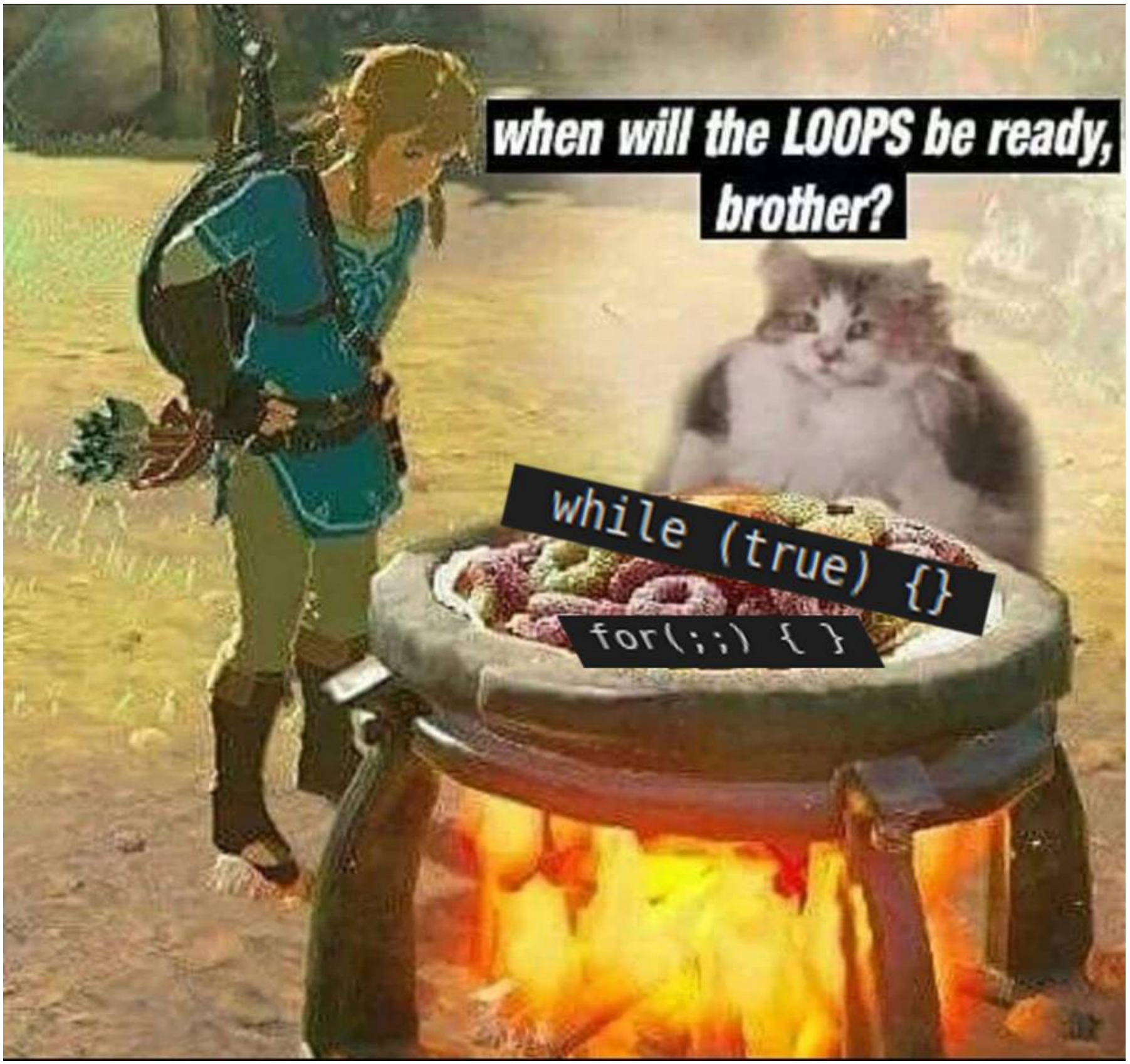
```
from turtle import forward, left


turtle_moves = [100, 200, 100, -20]
for move in turtle_moves:
    forward(move)
    left(90)
```

The infinite loop

```
while True:
    ...
```

Seen on SWU memes



General Feedback

What should be the name of your hand-in in case your ITU login is `abcd`?

- `abcd@itu.dk.zip (mailto:%60abcd@itu.dk.zip)`
- `group_with_funny_name.zip`
- `Wiglaf.txt`
- `abcd.zip`

Group Names

There are humans reading your assignments. Choose your group names so that you would not have a problem saying it when having coffee and cake with your grandparents on a Sunday afternoon.

Hand-in format.

The ZIP file contains:

```
├── A.py
├── B.py
├── C.py
└── D.py
```

And not:

```
├── A.zip
├── B.py
├── C.py
└── Another.zip
```

I found some code online...

Great! Take it, read it, understand it, modify it according to your use case and use the modified code.

Do not just copy and paste code without being completely sure what it does.

Pseudocode is almost code

In case you are in doubt, write pseudocode instead of code. It helps you organize and express your thoughts and it might allow the TA's to give more valuable feedback as they might understand better what you are trying to do.

In the same style, it is a good idea to write in comments where exactly you are lost and what you would like to do there.