MATH20622 Programming with Python - Lab test procedure

Posting of the problems

The test problems will be posted on Friday's during the lab class on the course website:

http://personalpages.manchester.ac.uk/staff/stefan.guettel/py/

As soon the problems become available, you can start working on them. Note that this is an examination and hence the examination policies apply:

http://www.tlso.manchester.ac.uk/map/teachinglearningassessment/assessment/sectiond-theprocessofassessment/policyonexaminations/

Note in particular that you need to be present in the correct lab class for doing the test (indicated on your time table), and you must sign the attendance sheet. It is not possible to "resit" a test or do it remotely. This is for fairness: all students must sit the test under the exact same conditions.

You can use all the materials on the course website, your previously written codes, as well as internet resources. You are not allowed to discuss with fellow students or ask for help from the lab assistants during the test. Mobile phones are not permitted, please put them away!

Structure of the test

The first "proper" lab test will take place on the Friday of week 4. The relevant material for that test will be the one from weeks 1, 2, and 3, including the exercises and lab class exercises posted on the course website.

There will be two problems, each of which will be solved in form of functions. Make sure to

- Start by creating **a single fresh `.py` file for each test** (for example, in the week 4 test you can call it `week4_yourname.py`). Save this file on your desktop and continue saving it repeatedly as you work on the problems to avoid data loss! Unfortunately, if your code is lost you can't submit it and hence won't get any marks for it.
- **Copy and paste the template provided on top of the test to your `.py`-file and fill out your personal details.** Double check your details, in particular, the email address. If the email address is incorrect, you cannot receive your scores and feedback.
- **Read the problem description carefully** before you start coding. The description will tell you precisely what your functions should do.
- **Start with the important core of each function,** i.e., the core code that solves each problem. You will be given marks for the functionality of your functions alone, hence, do not initially lose time on formatting your inputs or outputs.
- **Adhere to the provided template,** in particular, make sure that all code is within functions and not outside. Also, **avoid the use of `input()` calls** as these block the automatic execution of the codes. You can use fixed-valued variables for the testing.

- **Re-read the problem fully when you think you're done.** It's upsetting to have a well-written code but it fails to behave correctly just because a small detail was missed. Only functions that behave exactly as specified can achieve full marks. This is like in real life: You don't want your mobile phone software to crash just because the programmers have not accounted for all eventualities!
- **Do not chat with fellow students** until everyone's work has been submitted.

Submission

The lab test needs to be completed in form of a single `.py` file and it will be submitted via the Blackboard system. Multiple submissions are possible, but only the last submission will be marked. The strict deadline for submission is 10:50am for lab class A, and 14:50pm for lab class B. Submissions after this deadline or via channels other than Blackboard (e.g., Email) cannot be accepted.

Make sure to use the template provided with the test problems and that your personal details are correct. No code should appear outside the functions and the `input()` command should be avoided.

Marking

All codes will be marked automatically by testing various input-output combinations of the functions. This is why **it is important to adhere exactly to the function names and behaviour as specified on the test description.** Make sure that your code runs without crashing, as otherwise all marks may be lost. It is preferred to submit a non-crashing code which works partially over submitting a code that cannot be executed at all.

Feedback

The grading system will generate a feedback report for your submission which will then be emailed to you, provided you have specified a correct email address in the submitted code. If you have any further questions concerning the code assessment, or any other questions related to programming, please ask the helpers in the lab class.

Final words

These regular in-class tests are for you to check your understanding and to make sure to be on track with the course material. I hope you will find them interesting and challenging, but not annoying! All this time you invest into practising Python during the semester will hopefully pay off at the end as you should find completing the final coursework much easier.