EPCC

The University of Edinburgh
James Clerk Maxwell Building
Mayfield Road
Edinburgh EH9 3JZ
UK

Fax +44 131 650 6555
Telephone +44 131 650 5030
Direct line +44 131 650 6031
E-mail msc@epcc.ed.ac.uk

Dear MSc applicant,

On the following page you will find a programming exercise that you must attempt before your interview. You should email us full source code and screen output for each of the questions. The deadline for submitting your solutions is 48 hours before the interview. We will ask you some questions about your answers, so please ensure you have a copy available during the interview itself.

Even if you have not fully completed a question, we would rather you submitted incomplete code than no code at all. We are keen to see how you approached each problem even if you did not arrive at a full solution.

You **must** email us your answers as plain text in the main body of the message — we **will not** accept any form of attachments. Your email should take the form:

```
Name: <Your Name>

--------- Exercise 1 ---------
<Source code for Exercise 1>
<Screen output for Exercise 1>
--------- Exercise 2 ---------
<Source code for Exercise 2>
<Screen output for Exercise 2>
--------- Exercise 3 ---------
<Source code for Exercise 3>
<Screen output for Exercise 3>
<Explanation of Exercise 3>
-----------------------------
```

If you re-use any code between different exercises then you need only include it once.

Due to the number of people being interviewed we cannot enter into any correspondence regarding the exercises. However, if you are unclear about any part of a question then you should indicate in your answer what assumptions you have made and why.

Yours faithfully,

Dr. David Henty,
Programme Director,
MSc in HPC.

|epcc|

Programming Exercise for MSc Interviewees

Background

The following expansion gives an approximation to the exact value of π

$$\pi(N) = \frac{4}{N} \sum_{i=1}^{N} \frac{1}{1 + \left(\frac{i-\frac{1}{2}}{N}\right)^2}$$

For example, it is easy to check by hand that

$$\pi(1) = 4\frac{4}{5} = 3.2, \quad \pi(2) = 2\left(\frac{16}{17} + \frac{16}{25}\right) = 3.162\ldots.$$

It can be shown that the approximation continues to become more accurate as N is increased.

Exercises

Note that you must use double-precision variables for ALL floating-point numbers.

1. Write a program in C, C++, Fortran or Java that computes an approximation to π using the above formula for the following values of N: 1, 2, 10, 50, 100, 500. For each value of N, print out the approximate value $\pi(N)$ and the error $err(N)$. The error is the difference between $\pi(N)$ and the true value of π, ie $err(N) = \pi(N) - \pi$. As N increases the value of the error should decrease.
2. We now want to find out the *minimum* value of N that is required to give a value for $\pi(N)$ that is accurate to some specified value. We will call this value N_{min}. By computing $\pi(N)$ for increasing values of N, calculate N_{min} such that $err(N_{min}) < 10^{-6}$.
3. This way of computing N_{min} is clearly inefficient. For example, if we require $err(N_{min}) < 10^{-6}$ and we calculate $err(2) = 0.02$, it is a waste of time to calculate $err(3)$ as it is already obvious that N_{min} is very much larger than 2!

   Rewrite your program so that is uses a more efficient way to locate the minimum value of N. Your new method must produce the same value for N_{min} as before, but should run faster. For example, you might try and reduce the number of times that you have to evaluate $err(N)$. You should explain how your new program works and state how much faster it is than the one you wrote for Exercise 2. Does the increase in speed vary depending on the accuracy that is required? If so, can you explain this variation?