Queen Mary
University of London

Main Examination period 2019

MTH6150: Numerical Computing in C and C++

Duration: 2 hours

Apart from this page, you are not permitted to read the contents of this question paper until instructed to do so by an invigilator.

Log into your computer and start Visual Studio before the exam begins.

You should attempt ALL questions. Marks available are shown next to the questions.

Only non-programmable calculators that have been approved from the college list of non-programmable calculators are permitted in this examination. Please state on your answer book the name and type of machine used.

This is an OPEN BOOK exam.
Permitted:

- any printed material, e.g. books;
- any handwritten notes or photocopies;
- a memory stick or portable drive;
- use of a computer and the internet (QMPlus, google, wikipedia,...);
- access to your university network folder.

Prohibited:

- electronic communication devices (e.g. mobile phones, smart watches);
- any use of email;
- communicating via the internet (Facebook, Twitter, Stack Exchange,...);
- file sharing websites such as Dropbox;
- sharing material with other students.

Details of all internet activity on your computer may be logged.

To submit your solution there is a link on the module's QMPlus web page named "Exam submission". Upload the code files (with .cpp extension) here. There should be one file for each exam question.

Exam papers must not be removed from the examination room.

Examiners: J. Griffin, V. Nicosia

© Queen Mary University of London (2019)

All answers are to be written in C++ code files which will compile when included in a project. Use one file for each question. If a question asks for any statement about the results, include the answer as a comment in the code.

Question 1. [26 marks]
Suppose that we want to calculate

$$g(x;n) = \sum_{k=0}^{n-1} \frac{2^k \, cos(2(n-k)x)}{(k+1)!}$$

for any real x and positive integer n.

(a) Write a function that returns $g(x;n)$. The input arguments should be x and n. **[12]**

(b) Write code for an object which has n as a member variable, and a member function of the form shown below

```
struct Cosn{
    ...
    double operator()(const double x){
        ...
    }
};
```

This member function should take x as the input argument and return $g(x;n)$. Also give the object a constructor to assign a value to n. **[8]**

(c) Calculate and output to the screen $g(x;n)$ for $n = 12$ and $x = 2.6$, using the code from part (a) and then from part (b). **[6]**

Question 2. [18 marks]
Suppose that the size y of a population of animals at time t follows the following ordinary differential equation

$$\frac{dy}{dt} = \alpha + \beta y - \mu \;\left(1 + \varepsilon \, sin(2\pi t)\right) y$$

where α, β, μ, and ε are constants.
Use the Runge-Kutta fourth order method to numerically solve this equation from $t = 0$ to 20, with $y(0) = 200$, $\alpha = 200$, $\beta = 0.4$, $\mu = 0.6$ and $\varepsilon = 0.25$, using $n = 1000$ steps. Output the final value of $y(t)$ to the screen. **[18]**

© Queen Mary University of London (2019)

Question 3. [34 marks]
The standard logistic distribution has cumulative distribution function (cdf)

$$F(x) = \frac{1}{1+e^{-x}}, \; x \in \mathbb{R}.$$

(a) Write code that calculates the inverse function of $F(x)$, i.e. the function $G(u)$ such that if $u = F(x)$ then $x = G(u)$.

Hence, using the inverse cdf method, write code to generate random variables from this logistic distribution. **[12]**

(b) Using the code from part (a), fill in a vector with $10,000$ random values from the logistic distribution. Calculate the mean and variance of this random sample. **[10]**

(c) The above logistic distribution has probability density function

$$f(x) = \frac{e^{-x}}{(1+e^{-x})^2}, \; x \in \mathbb{R}.$$

Suppose that we want to evaluate the integral

$$I = \int_{-\infty}^{\infty} \frac{cos(x)\, e^{-x}}{(1+e^{-x})^2}\, dx.$$

Calculate an approximation to I using importance sampling, by generating a random sample of size $10,000$ from the logistic distribution. **[12]**

Question 4. [22 marks]

(a) Write a function called `contains` which takes as input arguments a `vector<int>` v and an `int` a. The function should return a `bool` value of true if any element of v is equal to a, and false otherwise. **[9]**

(b) Write another version of the function from part (a), this time as a template function called `contains_template` that takes as input a vector containing any numerical data type. The data type of the value to search for should be the same as the data type of the vector elements. **[6]**

(c) Use each of the functions from parts (a) and (b) to check whether the vector of elements $(7, 4, 2, 11)$ contains the value 4, using variables of type `int`.

Then use the function from part (b) to perform the same check, but using variables of type `unsigned long long`. **[7]**

End of Paper.

© **Queen Mary University of London (2019)**