MIDTERM EXAM 2019: COMPUTER SCIENCE II (B.MATH. 2ND YEAR)

- *This is a pen-and-paper exam, use of computers/calculators is not allowed.*
- *This is a closed book exam.*
- *All questions are compulsory. The question paper is of* 64 *points. The maximum you can score is* 50 *(that is, the scores above* 50 *are counted as* 50*).*

NOTE: During coding if you want to use a `SageMath` *inbuilt function and have forgotten the exact function name or order of parameters, use a made-up-name. State both the purpose of the function (referred by the made-up-name) and parameters it is taking (and what they do) separately and clearly. Important: you can use a different name for existing function, and not invent a function.*

Problem 1 (1.5 points each)**.** Answer the following in True or False. For a False answer, you need to give a counter example or an explanation as well. For a True answer, you can just mention it is True.

(1) `SageMath` works with fixed precision numbers, by default fixed to 53 bits – in particular, output of a command like `factorial(100)` is accurate only to 53 significant bits.
(2) The command `print [1,2]+[3,4]` produces `[1,2,3,4]` as output.
(3) Working with real numbers of precision 4 bits in `SageMath` (that is, defined via `RealField(4)`), the number 1.0001 (in binary) will be rounded to 1.001.
(4) Newton-Raphson method cannot be applied to complex valued functions becuase there is no fixed tangent line to a complex surface.
(5) It is better to use the iterative calculation $x_{n+1} = x_n - \frac{f(x_n)}{\frac{f(x_n)-f(x_{n-1})}{x_n - x_{n-1}}}$ over $x_{n+1} = \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n)-f(x_{n-1})}$ since the latter can accumulate cancellation errors.
(6) For an invertible matrix A, it's condition number $\kappa = \|A\| \cdot \|A^{-1}\|$ is always greater than or equal to 1.
(7) Let A be a $n \times n$ square matrix and let b be a $n \times 1$ column vector. Let $B = [A\ b]$ be the $n \times (n+1)$ block matrix. Then $rank(B) \geq rank(A)$ and equality holds if and only if A is invertible.
(8) In using Netwon's method to compute simultaneous zeros of several non-linear functions $\underline{f} = (f_1, \cdots, f_n)$ in several variables $\underline{x} = (x^{(1)}, \cdots, x^{(m)})$ the number of variables m must be equal to the number of equations n because we need to use the *Jacobian* matrix which must be a square matrix so that we can write down it's inverse.

Problem 2 (5 marks)**.** The following `SageMath` function takes as input an integer `n` and is supposed to output `n`-th Fibonacci number, however it is incorrect. Correct the typo and explain why it caused a problem.

```
def fibonacci(n):
    xp = 0
    x = 1
    for i in range(0,n):
        tmp = x
        x = x + xp
        xp = tmp

        return xp
```

Problem 3 (8 marks)**.** Let $f(x) = \ln(1+x)$ (here ln denotes natural logarithm, that is log to base e). Write down it's Taylor series expansion around $x = 0$. Till how many terms should we sum the Taylor series to calculate $f(1/2)$ to an accuracy of 53 bits?

Problem 4 (8 marks)**.** Consider the quadratic equation $ax^2 + bx + c = 0$. It's solutions are given by

$$r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{2c}{-b - \sqrt{b^2 - 4ac}}$$
$$r_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} = \frac{2c}{-b + \sqrt{b^2 - 4ac}}.$$

Which of the two expressions would you use to compute r_1 and r_2 (assume that $ac \ll b^2$)? Explain why.

Problem 5 (8 marks)**.** Find a bound for the number of iterations of Newton-Raphson method needed to achieve an approximation with accuracy 2^{-8} to the solution of $x^3 + x - 3 = 0$ lying in interval the $[1, 3]$. What is the numbers of steps, were we to work with bisection method instead?

Problem 6 (8 marks)**.** Let $g : [a, b] \to [a, b]$ be continously differentiable, and $|g'(x)| \leq c < 1$ for $x \in (a, b)$. Show that g has exactly one fixed point in the interval $[a, b]$ and it can be obtained as the limit of iterative sequence of fixed-point method, $x_0 \in (a, b)$ arbitrary, $x_{n+1} := g(x_n)$ for $n \geq 0$.

Problem 7 (15 marks)**.** Write down a `SageMath` function which takes as input a 3×3 matrix and produces LU decomposition of the given matrix (you can demand the 3×3 input matrix as a `SageMath` `matrix`, as a `Python` list of lists, or in any reasonable form that suits you). Do NOT USE the inbuilt function `matrix.LU` of `SageMath`.

—THE END—