NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER I EXAMINATION 2011-2012

MTH 213 – Experimental Mathematics

December 2011 TIME ALLOWED: 2 HOURS

INSTRUCTIONS TO CANDIDATES

1. This examination paper contains **FIVE (5)** questions and comprises **FOUR (4)** printed pages.
2. Answer all questions. The marks for each question are indicated at the beginning of each question.
3. Answer each question beginning on a **FRESH** page of the answer book.
4. This **IS NOT an OPEN BOOK** exam.
5. Candidates may use calculators. However, they should write down systematically the steps in the workings.

Question 1. (20 marks)

(i) Write a function `func1` that takes as input a list of 3 numbers $[a_2, a_1, a_0]$ and returns the polynomial $a_2x^2 + a_1x + a_0$ which has these numbers as coefficients.
For example, `func1([1, 2, 3])` should return `x^2 + 2*x + 3`.

(ii) Write a function `func2` which accepts a list $[a_{n-1}, a_{n-2}, \ldots, a_1, a_0]$ for any length n, and a number $k \geq 0$ and returns the k-th derivative of the polynomial $a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0$.

Question 2. (20 marks)

Carol wants to compute the Taylor series of a function $f(x)$ around the point $x = 1$, up to the n-th term. The first term is clearly the value $f(1)$. Carol implemented a function `get_taylor_coeff` to compute the Taylor coefficients. But this function is giving incorrect answers. Find the error(s) in the function and correct the error(s).

```
# Get the taylor coefficients of the function f(x) upto the
# k-th term around the point x0.
# For k=1, it should return [f(x0)].
def get_taylor_coeff(f, x0, k):
    coeffs = []
    for i in srange(1, k):
        t(x) = f.derivative(x, i)
        coeffs.append(t(i))
    return coeffs

# Example usage of get_taylor_coeff
f(x) = x^20 - x^4 + 1
get_taylor_coeff(f, 1, 3)
```

Question 3. (20 marks)

Consider the following interpolation problem. Let

$$p(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0$$

be a polynomial. The graph of the corresponding function $x \mapsto p(x)$ passes through the points $(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$. Adam wrote the following Sage code to return the list $[a_{n-1}, \ldots, a_0]$ of coefficients of the polynomial $p(x)$. Here, `ylist` is the list $[y_0, y_1, \ldots, y_{n-1}]$ and `xlist` is the list $[x_0, x_1, \ldots, x_{n-1}]$.

```
def get_coeff(xlist, ylist):
    n = len(xlist)
    def f(i, j):
        return xlist[i]^j
    M = matrix(RDF, n, n, f)
    yvec = vector(RDF, ylist)
    return M.solve_right(yvec)
```

(i) Adam is getting an incorrect answer from `get_coeff` when he is trying to get the coefficients of a degree two polynomial which passes through the points $(1, 5), (2, 10), (3, 17)$. Find the error(s) in the function `get_coeff` due to which Adam is getting the incorrect answer. Give the corrections.

(ii) Is there any degree 2 polynomial $p(x)$ for which the *incorrect* function `get_coeff` would still give a correct solution? If so, then give an example of such a polynomial. If such a polynomial cannot be obtained, explain why this is the case.

Question 4. (20 marks)

Eve decided to compute a certain function using recursion. The function Eve wrote is the following:

```
def compute(a, b):
    if a < 0 or b < 0 or a < b:
        return 0
    if b == 0:
        return 1
    return a*compute(a-1, b-1)/b
```

(i) What mathematical function is Eve's function `compute()` evaluating?

(ii) Write a non-recursive version of Eve's function `compute()`.

Question 5. (20 marks)

Let π be a permutation of the set $I = \{0, \ldots, n-1\}$. The *orbits* of π on I are the equivalence classes of the binary relation $\equiv_\pi$ on I, so that $x \equiv_\pi y$ if and only if there exist $i \geq 0$ such that $\pi^i(x) = y$. Here π^i denotes the i-th iteration of π, i.e. $\pi^0(x) = x$, $\pi^1(x) = \pi(x)$, $\pi^2(x) = \pi(\pi(x))$, etc. Write a Sage function that takes π as a list of length n of numbers in I and returns the list of lengths of the orbits of π on I.

END OF PAPER