ENGR 244 – ENGINEERING COMPUTING – FALL 2017 – TERM EXAM II

Please read the following information closely.

- You have **110 minutes** from the moment that the beginning of the exam is announced.

- This exam consists of five problems. Each is worth up to 14 credits, out of 100 credits for the whole course. At most 35 credits can be gained from the present term exam.

  You need to **work on three problems**, at least, to achieve an optimal outcome.

- If you **choose to work on three problems**, the two problems with the best outcome will count normally (i.e., up to 14 credits each), and the third problem will count with a factor of 50% (i.e., up to seven credits), yielding an **optimum total of 35 credits**.

- If you choose to work on more than three problems, the outcomes will be ordered by the number of credits achieved. **The two best problems count with a factor of 100%**, and **the third problem counts with a factor of 50%**, yielding up to 35 credits for the exam as a whole. The outcome of the other problems does not influence your grade.

- Appealing to the head of department, board of trustees, etc., will not change any grades.

- If you use **handwritten notes**, they need to be turned in as part of your exam material.

  (They will not be returned to you.)

The exam time is greater than the time that should be required. Recall that it is sufficient to **solve three out of the five present exam problems**. Accordingly, feel free to hand in your submission at any time and leave the room quietly without disturbing the other participants.

Any access to means of communication is a case of cheating irrespective of what is communicated. It is enough to **turn off your cell phones**. There is no need to place them on the front desk.

In your own interest, please avoid unnecessarily long answers.

AUIS student ID number: _____

Full name: _____

TERM EXAM II – PROBLEM 1

Characterize, as exactly and concisely as possible, **the outcome** of the following program:

```
#include <fstream>
#include <iostream>
#include <string>

using namespace std;

int main( )
{
  ofstream fileout("AUIS");

  char E = 'E';
  char z = 'z';
  int a = z + z;

  string N = "NGR ";
  if(E > z) fileout << "AUIS\n";
  else fileout << E + N << a << '\n';

  fileout.close();
  return 0;
}
```

Characters can be used for integer arithmetics. The relevant parts of the **ASCII character table** are given below; first: interpretation as text, e.g., 'C'; second: interpretation as a number, e.g., 67.

American Standard Code for Information Interchange (ASCII)

'0'	48	'A'	65	'a'	97
'1'	49	'B'	66	'b'	98
'2'	50	'C'	67	'c'	99
'3'	51	'D'	68	'd'	100
'4'	52	'E'	69	'e'	101
'5'	53	'F'	70	'f'	102
...	...	...	...	...	...
'9'	57	'Z'	90	'z'	122

As discussed during the lecture, adding a character to a character will follow the rules of integer arithmetics, whereas adding a string to a string, or a character to a string, combines them as text.

In your own interest, please avoid unnecessarily long answers.

TERM EXAM II – PROBLEM 2

Note: *You can get a substantial fraction of the credits here without writing actual C/C++ code, but by correctly describing what the program should do ("pseudocode" or step-by-step description of the performed arithmetics). If you cannot solve* **part a)**, *make* **reasonable assumptions** *on what a solution must look like to answer* **part b)**.

a) Write a function called "next_leap_year" with a single argument, **int** *n*, and a return value which is specified as follows: The argument *n* represents a year of the standard Gregorian calendar, and **leap years** are defined according to the standard Gregorian calendar. The return value of

next_leap_year(*n*)

is given by *n* itself if the year *n* is a leap year, and otherwise by the smallest integer $m > n$ such that the year *m* is a leap year. For example, 2017 is not a leap year, neither are 2018 and 2019, but 2020 is; hence, next_leap_year(2017) and next_leap_year(2020) should both return 2020.

By Gregorian convention, **a year *n* is a common year if** it fulfills any of these two conditions:

- *n* is **not divisible by 4** (i.e., its remainder after division by 4 is not 0), or if
- *n* is **divisible by 100** (and hence also by 4), but **not divisible by 400**.

  (If you are unsure how to implement this, continue by using the first condition only.)

A year is a leap year if it fulfills none of these two conditions, i.e., **if it is not a common year**.

Year zero does not exist (hence, never return 0); it is up to you how you treat the case $n < 0$.

b) Discussion of time requirements: Consider each assignment, each arithmetic operation, and each logical operation as one elementary operation. For the function that you wrote above, how does the number of elementary operations scale asymptotically for great values of *n*?

Explain **briefly** how you obtain the result. (You **do not need to give the exact number of operations**, only the order of its dependence on *n*, as usual for discussions of **asymptotic scaling**.)

In your own interest, please avoid unnecessarily long answers.

TERM EXAM II – PROBLEM 3

Characterize the return value of the following **recursive function**.

```
long double exp_recur(double m, int n)
{
  if(n < 0) return 0.0;

  double numer = 1.0;
  for(int i = 0; i < n; i++) numer = numer * m;  // compute: m to the power of n

  double denom = 1.0;
  for(int i = 1; n >= i; i++) denom = denom * i;  // compute: n factorial

  return exp_recur(m, n - 1) + numer/denom;
}
```

a) What is the return value of exp_recur(1, -1), i.e., using the argument values $m = 1$ and $n = -1$?

b) What is the return value of exp_recur(1, 2)?

c) Deduce a **general** expression for the **return value** of exp_recur(1, n), for natural numbers $n \geq 1$.

Assume here that the return value of exp_recur(1, n - 1) is given by $\sum_{k=0}^{n-1} \frac{1}{k!}$.

d) What is the return value of exp_recur(m, 1), for any m, using the argument value $n = 1$?

e) Give a **general** expression for the **return value** of exp_recur(m, n), for any m, with $n \geq 0$.

Note: The data type **long double** is just like **double**, but with a greater precision.

In your own interest, please avoid unnecessarily long answers.

TERM EXAM II – PROBLEM 4

Consider the program given below.

```
#include <iostream>
using namespace std;

void xcube_cout(double x, double last_x, int rows)
{
  double delta_x = (last_x - x) / (rows - 1.0);
  cout << "Cubic values will be written to standard output.\n\n";

  for(int i = 0; i < rows; i++)  // loop to be analysed here
  {
    cout << x << "  ";
    cout << x * x * x << "\n";
    x += delta_x;
  }
  return;
}

int main( )
{
  double x = 2.0;
  double y = 4.0;

  // state A (before calling xcube_cout)

  xcube_cout(x, y, 3);

  // state B (after xcube_cout has returned)

  return 0;
}
```

a) What is the output of this program, i.e., what is written to **standard output** (stream "cout")?

b) What is the **return type** of the function "xcube_cout"?

c) Is this a case of passing arguments *by value* or *by reference*? What is the value of *x* in the function "main" in **state A** (before calling "xcube_cout") and in **state B** (after "xcube_cout" returns")?

d) Consider the **for** loop in the function "xcube_cout", i.e., the loop which is marked with the comment *"loop to be analysed here"*. What is the **precondition** of this loop, what is its **loop condition**, what is its loop **invariant**, and what is its **postcondition**?

In your own interest, please avoid unnecessarily long answers.

TERM EXAM II – PROBLEM 5

Note: *You can get a substantial fraction of the credits here without writing actual C/C++ code, but by correctly describing what the program should do ("pseudocode" or step-by-step description of the performed arithmetics).*
If you cannot solve **part a)**, *make* **reasonable assumptions** *on what a solution must look like to answer* **part b)**.

a) Write a function which, to a sufficient level of accuracy, computes and returns the value of

$$\sum_{k=0}^{\infty} \frac{x^k}{k!},$$

wherein floating-point arithmetics are used, and **double** *x* is the single argument of the function.

b) Make a statement on the reached **level of accuracy**; why do you believe that this is sufficient?

In your own interest, please avoid unnecessarily long answers.

ENGR 244 – ENGINEERING COMPUTING – FALL 2017 – TERM EXAM II

AUIS student ID number: _____

Full name: _____