CS 58000_01 Algorithm Design, Analysis & Implementation (3 cr.)

Assignment As_01

Student Name:_____

This assignment As_01 is due at 12:00 midnight, Monday, February 12, 2022. Please submit your assignment to Brightspace (purdue.brightspace.com). No late turn-in is accepted. Please write your name on the first page of your assignment (5 points off if no name is given). Your file name should be your last name such as JCarr_CS486_As01.docx. (5 points off if fail in a correct filename).

The total points for this assignment As_01 are 80 points.

I: [80 points]: each of I.1 through I.8 is 10 points.

Given two algorithms "function divide(x, y)" in my lecture note Ch 00_02_S_IntroFoundation_ProgCorrectionLec.pptx, (posted on purdue.brightspace.com), the first one is iterative and the second one is recursive. The operators used are: "shift right one bit", "shift left one bit", copy, add (+), compare two contents of given variables (such as ≥), assign (:=), and call statement (such as a recursive call statement).

Algorithm a:

function divide(x, y)

Input: Two integers x and y, where y ≥ 1.

Output: The quotient and remainder of x divided by y.

```
if  x = 0, then return (q, r) := (0, 0);
        q := 0;  r := x;
while (r ≥ y) do
        { q := q + 1;
        r := r – y};
return (q, r);
```

Algorithm b:

function divide(x, y)

Input: Two integers x and y, where $y \geq 1$.

Output: The quotient and remainder of x divided by y.

```
if (x = 0) then return (q, r):=(0, 0);
(q, r) := divide(⌊x/2⌋, y )
q := 2 * q,  r := 2 * r;
if (x is odd) then r := r + 1;
if (r ≥ y) then
        { r := r – y; q := q + 1};
return (q, r);
```

(I.1) Give the input and output specifications for :

(a) the algorithm a?
(b) the algorithm b?

(I.2) What is the input size for:

(a) the algorithm a?
(b) the algorithm b?

(I.3) What is the basic operation for :

(a) the algorithm a?
(b) the algorithm b?

(I.4) In algorithm b, what is the functionality of the following segment of statements? (i.e., give the reasons for writing each of the statements. Why double the q and r? Why increase r by one when x is odd? Why reduce r by y and increase q by one when $r \geq y$?)

```
q := 2 * q;  r := 2 * r;
if (x is odd) then r := r + 1;
if (r ≥ y) then
        { r := r – y; q := q + 1};
```

(I.5) Analyze and derive the algorithm's time and space efficiency for Algorithm a.

(Hint: express time efficiency in terms of summation $\sum$)

Given algorithm a,

```
if  x = 0, then return (q, r) := (0, 0);
        q := 0;  r := x;
while (r ≥ y) do       // takes n iterations for the worse case.
        { q := q + 1;
        r := r – y};  // O(n) for each r – y, where y is n bits long.
return (q, r);
```

(I.6) Analyze and derive the algorithm's time and space efficiency for algorithm b.

(Hint: for time efficiency, use recurrence relation and solve for the recurrence relation system)

Given algorithm b,

```
if (x = 0) then return (q, r):=(0, 0);
(q, r) := divide(⌊x/2⌋, y )        //requires n-bits right shift
q := 2 * q,  r := 2 * r;           // shift left one bit.
if (x is odd) then r := r + 1;     // needs c*n-bits
if (r ≥ y) then                    // additions
        { r := r – y; q := q + 1};
return (q, r);
```

(I.7) Can these two algorithms a and b be improved? Justify your answer.

(a) for the algorithm a?
(b) for the algorithm b?

(I.8) Compare these two algorithms a and b:
Which is a better algorithm in terms of time and space efficiency? Justify your answer.

Note: If you provide your answer in your handwriting, good handwriting is required.
Proper numbering of your answer to each problem is strictly required.
The solutions to the problems must be orderly given. (10 points off if not)