Name: Jiaxin Li(jli2274@wisc.edu)

Sunjun Gu(sgu59@wisc.edu)

Assignment: hw02

1. (a)

**Algorithm**

**Input:** An integer n, n>=0

**Output:** The binary representation of

1: **procedure** PowTen(n)

2: **if** n=0 **then**

3: **return**

4: **else if** n=1 **then**

5: **return**

4: **else**

5:

6:

7: s

10: **if**

11:

12: **else**

13:

\* is the same with algorithm in the lecture notes with the same name

where

Input: Two numbers a and b, in binary, two n-bit integers.

Output: The number a·b, in binary

Runtime:

**Proof of Correctness:**

The proposition here P(n) is that this algorithm can return the correct binary representation of given the input integer n (n≥0) which is the power exponent of the decimal number .

**Base case:** When n=0, the decimal number is 1; n =1, the decimal number is 10. Check the *Decimal and Binary Comparison Table*, . Therefore, P(0), P(1) is correct.

**Induction Step:** By strong induction,suppose P(2), … , P(k-1), P(k), (k >1,k∈Z) is correct. Such that given a decimal number with the power exponent k, the algorithm can return the correct binary representation of .

To prove the correctness of P(k+1):

The given decimal number  has the power exponent k+1.

k>1 🡪 k+1>2 🡪 n>2

so line 5～line 13 will be executed.

Line 5: m =  =

Line 6: Because k >1,k∈Z 🡪 2k>k+1 🡪 k >, and is applied the floor function. Therefore, 0 ≤ m = ≤ < k and P(m) is correct under the previous assumptions. c = P(m) is the binary representation of .

Line 7: Apply the algorithm on c, get the number c·c in binary, marking as s.

Line 10～11: When n is even, namely k+1 is even, we can write  = = , that is because when k+1 is even，m = = ，m+m = = = k+1; Therefore, we could see the binary representation of as the binary representation of . From line 6, we get the binary representation of , c = P(m). From line 7, we get the binary integer multiplication of c and c, s =. Therefore, s is the binary representation of .

Line 12～13: When n is odd, namely k+1 is odd, we can write = = , that is because m = = , = . Therefore, we could see the binary representation of as the binary representation of . From line 6, we get the binary representation of , c = P(m). From line 7, we get the binary integer multiplication of c and c, s =. Check the *Decimal and Binary Comparison Table*, .Therefore, the binary representation of is s·（）= s· + s·. And this answer could be get by bit shifting of 3 to the left, bit shifting of 1 to the left, then adding them up.

Therefore, the binary representation of would be returned correctly by the algorithm. P(0),P(1),P(2),…,P(k)🡪P(k+1).

By mathematical inductive principle, P(n) is true for all n≥0.

**Proof of Time Complexity:**

We use the recursion tree to organize the calculation. Since the recursive case of the implementation makes two recursive calls (line 6 and line 7), the recursion tree is a binary tree.

CORRECT:

1. BC: n=0;
2. IS: n>1,[n/2] 🡪n even/odd🡪n

RUNTIME:n🡪n/2🡪n/4…1/cn^q🡪c(n/2)^q…c(n/2^d)^q,c’

1. (b)

**Algorithm**

**Input:**

**Output:**

1: **procedure** DecToBin(X)

2: **if** n=1 **then**

3: **return** check from table

4: **else**

5:

6:

7:

8:

9:

10:

3

**Algorithm**

**Input:**

**Output:**

1: **procedure** Knapsack(v, w, W, n)

2: **if** W < **then**

3: **return**

4: **else**

5:

6:

7:

8:

9: Add i into subset of items **Item[]**

10:

11: