Chapter 2 Homework

Due date: Nov. 9, 2017

Program Exercises

- 1. Write a function, *pmult*, that multiplies two polynomials, A(x) and B(x), to obtain D(x). Requirements:
 - a. There is an array to store A(x), B(x), and D(x). Each element of this array is composed of two fields, coefficient and exponent, to store each term of non-zero coefficient of A(x), B(x), and D(x). The index of the first term of A and B is given by startA and startB, respectively. finishA and finishB give the last term of A and B. The index of the next free location is given by available. *startD and *finshD are pointers point the starting and ending locations of D in the array.
 - b. Create such an array for A(x), B(x), and D(x).
 - c. Use startA, startB, finishA, finish, *startD and *finshD as function inputs.
 - d. Print out the array as output
- 2. Rewrite *fastTranspose* so that it uses only one array rather than two arrays required to hold *rowTerm* and *startingPos*.

Requirements:

a. You may use the 6×6 sparse matrix as shown in the following as input to check your program.

	row	col	value
a[0]	6	6	8
[1]	0	0	15
[2]	0	3	22
[3]	0	5	-15
[4]	1	1	11
[5]	1	2	3
[6]	2	3	-6
[7]	4	0	91
[8]	5	2	28

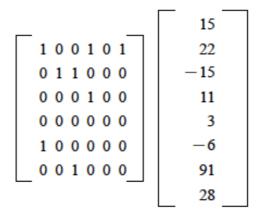
- b. Print out the transpose matrix in above form.
- 3. Write a function *inverse* to inverse a sparse matrix.

Requirements:

- a. Use the sparse matrix *a* (as shown in Exercise 2a) as an input. Create a matrix called *b* (of the same size of *a*) as another input. The function *inverse* should inverse *a* and store the result in *b*.
- b. Print out b if the inverse matrix of a exists; otherwise, print out a string saying that "The

matrix is non-invertible." Please note that b should be represented as the same form of a, and you can **only** implement *inverse* using the sparse matrix representation.

4. Another sparse matrix representation is to keep only the nonzero term in a one-dimensional array, called *value*, in the order described in the text. In addition, a two-dimensional array, called *bits*[rows][columns], such that *bits*[i][j]=0 if a[i][j]=0, and *bits*[i][j]=1 otherwise, as shown in the following.



Requirements:

- a. Denote above sparse matrix as *a*. Create another sparse matrix as *b* and you can insert any element values you want to create *b*.
- b. Write a C function to obtain d=a+b.
- 5. Modify the function *strinins* (string insertion function) so that it does not use a temporary string *temp*.

Requirements:

- a. Using strings *s* and *t*, and an integer *i* as input such that *t* is inserted to *s* starting at the *i*th position.
- b. Print out your s, t, and the string after the insertion.