Array:

1. Write a program which multiplies two polynomials represented using the array. What is the computing time of your procedure?

2. When all the elements either above or below the main diagonal of a square matrix are zero, then the matrix is said to be triangular.
   In a lower triangular matrix, A, with *n* rows, the maximum number of nonzero terms in row *i* is *i*. Hence the total number of nonzero terms is
   $\sum_{i=1}^{n} i = n(n+1)/2$. For large *n* it would be worthwhile to save the space taken by the zero entries in the upper triangle. Obtain an addressing formula for elements a_{ij} in the lower triangle if this lower triangle is stored by rows in an array *B[1…n((n+1)/2*] with *A[1,1]* being stored in *B[1]*. What is the relationship between *i* and *j* for elements in the zero part of A?

3. Let *A* and *B* be two lower triangular matrices, each with *n* rows. The total number of elements in the lower triangles is *n(n+1)*. Devise a scheme to represent both the triangles in an array *c [1...n, 1...n+1]*. [Hint: represent the triangle of *A* as the lower triangle of *c* and the transpose of *B* as the upper triangle of *c*.] Write a program to determine the values of A[i,j], B[i,j], $1 \le i, j \le n$ from the array c.

4. Another kind of sparse matrix that arises often in numerical analysis is the tridiagonal matrix. In this square matrix, all elements other than those on the major diagonal and on the diagonals immediately above and below this one are zero. If the elements in the band formed by these three diagonals are represented row-wise in an array *b* with *A[1,1]*, being stored in *b [1], write a program* to determine the values of *A[i,j]*, $1 \le i, j \le n$ from the array *b*.

5. An *m* x *n* matrix is said to have a *saddle point* if some entry *A[i,j]* is the smallest value in row *i* and the largest value in column j. Write a program which determines the location of a saddle point if one exists.

Linked lists :

1. Write a program *length* to count the number of nodes in a singly linked list *p* where *p* points to the first node in the list. The last node has link field **null**.
2. Let *p* be a pointer to the first node in a singly linked list and *x* a pointer to an arbitrary node in this list. Write a program to delete this node from the list. If *x = p,* then *p* should be reset to point to the new first node in the list.
3. Let $x=(x_1, x_2, \ldots, x_n)$ and $y=(y_1, y_2, \ldots, y_n)$ be two linked lists. Write an algorithm to merge the two lists together to obtain linked list $z=(x_1, y_1, x_2, y_2, \ldots, x_m, y_m, x_{m+1}, \ldots x_n)$ *if* $m \le n$ *and* $z=(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, y_{n+1}, \ldots y_m)$. If $m>n$. No additional notes may be used.
4. Do exercise 1 for the case of circularly linked list.
5. Do exercise 2 for the case of circularly linked list.
6. Do exercise 3 for the case of circularly linked list.
7. Devise a representation for a list where insertions and deletions can be made at either end. Such a structure is called deque. Write a program for inserting at either end.
8. Write a program for a singly linked circular List which reverses the direction of the links.
9. Let *p* be a pointer to the circularly linked list. Show how this list may be used as a queue. I.e., write a program to add and delete elements. Specify the value for *p* when the queue is empty.
10. Write a program *pread(x)* to read in *n* pairs of coefficients and exponents, (c_i, e_i) $1 \le i \le n$. of a unvaried polynomial *x* and to convert the polynomial into the circular linked list structure. Assume $e_i > e_{i+1}$, $1 \le i < n$ and that $c_i \neq 0, 1 \le i \le n$. Your program should leave *x* pointing to the head node.
11. Let *a* and *b* be pointers to the head nodes of two polynomials represented as in Exercise 10. Write a program to compute the product polynomial $c = a * b$. Your program should leave *a* and *b* unaltered and create *c* as a new list. Show that if *n* and *m* are the number of terms in *a* and *b* respectively, then this multiplication can be carried out in time or $O(nm^2)$ or $O(mn^2)$. If *a, b* are dense show that the multiplication takes $O(mn)$.
12. Let *a* be a pointer to the head node of a univariate polynomial . Write a program *peval (a,x)* to evaluate the polynomial *a* at the point *x,* where *x* is some real number.
13. Let *a* and *b* be two sparse matrices. Write a program *mmul(a,b,c)* to set up the structure for $c = a*b$. If *a* is an *n* x *m* matrix with r_A non-zero terms and if *b* is an *m* x *p* matrix with r_B with non-zero terms.
14. Write a program to write out the terms of a sparse matrix *a* as triples (i,j,a_{ij}) The terms are to be output by rows and within rows by columns.
15. Write an algorithm *mtrp(a,b)* to compute the matrix $b = a^T$, the transpose of the sparse matrix *a* .