4/11/22

__/__/____

Name : Krishnaraj P. T
PRN : 10322 10838
Roll : PA 20 Batch – A1
Div : SY CSF

FDS Theory Assignment 1

Q. 1. Write a program to represent a polynomial using an array of structures. Write an algorithm for polynomial multiplication.

→

```
#include <stdio.h>
struct Polynomial {
    int coeff;
    int exp;
};

int main() {
  int power = 1;
  printf("Enter the power of your polynomial");
  scanf("%d", &power);
  if (power == 0) return 0;
  struct Polynomial p[power + 1];

  printf("Enter your polynomial");
  for (int i = power; i >= 0; i--) {
    printf("Enter coeff with power %d", i);
    scanf("%d", &p[i].coeff);
    p[i].exp = i;
}

printf("Here is your polynomial with a
        conventional variable x");
```

```
printf ("\n");
for (int i = power ; i >=0 ; i--)
{
    if ( p[i].coeff != 0 )
    {
        if ( i != power )
        {
            printf (" + ");
        }
        printf (" %d x^ %d ", p[i].coeff , p[i].exp);
    }
}

return 0;
}
```

(*) Alg for polynomial multiplication

→

```
Algorithm Poly-multiplication ( p1, p2, max1, max2)
{
    i = j = k = 0
    while ( i < max1 ) : {
        j = 0
        while ( j < max2 ) : {
            temp = p1 [i].coef * p2 [j].coef
            if ( temp != 0 ) : {
                flag = 0 ;
                exp = p1 [i].exp + p2 [j].exp ;
                for (x=0 ; x<k ; x++) : {
```

```
            if (exp == p3[x].exp) {
                flag = 1
                break; }
        } // for loop

        if( flag == 1) {
            p3[x].coeff = p3[x].coef + temp;
            j++;
        }
        else {
        p3[k].exp = exp;
        p3[k].coeff = temp;
         j++ ; k++;
        }
      } // if
    } // while
    i++ ;
} // main while
```

Q.2. Sorting elements is ~~de~~ descending order, considering Salary as keys.

Salary	9000	8915	8800	9130	8000
Student Name	Arjun	Sanita	Mihir	Ruchir	Anvi.

→ Pass - 1 :

9000	8915	8800	9130	8000
Arjun	Sanita	Mihir	Ruchir	Anvi.

(No change)

(*) Pass - 2 (same)

9000	9915	8800	9130	8000
Arjun	Sanita	Mihir	Kuhir	Anvi

(*) Pass - 3

9000	8915	8800	9130	8000
Arjun	Sanita	Mihir	Kuhir	Anvi

~~Swap~~ move

(gives)

9130	9000	8915	8800	8000
Kuhir	Arjun	Sanita	Mihir	Anvi

(*) Pass - 4 (No change) [n-1 passes total]

Final sorted Array

9130	9000	8915	8800	8000
Kuhir	Arjun	Sanita	Mihir	Anvi

→ Insertion sort is stable; order of elements remains same

Q.3. 10, 20, 22, 26, 89, 90, 99, 100, 120 is sorted in ascending order.

Design an algorithm to insert 25 into given list.

(*) as array is sorted in ascending order, we can find an element greater than key, and insert key there. Then move all other elements to the right.

→ Swap (a, b) // Required

```
{   temp = a
    a = b
    b = temp
}
```

→ algorithm Insert ascending (arr , n , key) :

```
{
// Find where to insert.
pos = -1    // default.
i = 0
while ( arr [i] < key ) { }
    { i++ }
pos = i - 1
arr [n] = 0   // safety to avoid swapping wrong data
for ( i = n-1 ; i >= pos ; i-- )        given
{   swap ( arr [i+1], arr [i] )   // array is large enough.
}
arr [pos] = key ;   // insert element.
}.
```