Algorithm Lab 2

Supta Philip Philip

June 2022

1 Pre-Requisite in C++

1. Function
2. Pointer
3. Array / Dynamic Array
4. Dynamic Array - 2D/ Matrix
5. Random number
6. File Operation
7. Standard Template Library(STL)

2 Warm up problem

1. Fibonacci
2. Factorial
3. Prime number
4. Euclid's algorithm for GCD
5. Maximum Pairwise Product
6. Find second largest number in a List
7. Linear Search
8. Bubble sort
9. Selection sort
10. Insertion Sort
11. Counting Sort

Algorithm 1 Fibonacci

```
1: procedure FIBONACCI(n)
2:     if n ≤ 1 then
3:         return n
4:     else
5:         return FIBONACCI(n − 1)+FIBONACCI(n − 2)
6:     end if
7: end procedure
```

Algorithm 2 Factorial

```
1: procedure FACTORIAL(n)
2:     if n ≤ 0 then
3:         return 1
4:     else
5:         return n*FACTORIAL(n − 1)
6:     end if
7: end procedure
```

Algorithm 3 IsPrime

```
1: procedure ISPRIME(n)
2:     Flag ← true
3:     for i ← 2, √n do
4:         if n%i == 0 then
5:             Flag ← false
6:             break
7:         end if
8:     end for
9:     return Flag
10: end procedure
```

Algorithm 4 Euclid's algorithm for GCD

```
1: procedure GCD(a, b)                          ▷ The g.c.d. of a and b
2:     r ← a mod b
3:     while r ≠ 0 do                           ▷ We have the answer if r is 0
4:         a ← b
5:         b ← r
6:         r ← a mod b
7:     end while
8:     return b                                 ▷ The gcd is b
9: end procedure
```

Algorithm 5 Euclid's algorithm

```
1: procedure GCD(n, m)
2:     if n = m then
3:         return m
4:     else if n ≥ m then
5:         return GCD(n − m, m)
6:     else
7:         return GCD(n, m − n)
8:     end if
9: end procedure
```

Algorithm 6 Maximum Pairwise Product

```
1: procedure MaxPairwiseProductNaive(A, n)
2:     product ← 0
3:     for i ← 1, n do
4:         for j ← 1, n do
5:             if i! = j then
6:                 if product < A[i] ∗ A[j] then
7:                     product = A[i] * A[j]
8:                 end if
9:             end if
10:        end for
11:    end for
12:    return product
13: end procedure
```

Algorithm 7 Maximum Pairwise Product Fast

```
1: procedure MaxPairwiseProductFast(A, n)
2:     index_1 ← 0
3:     for i ← 0, n − 1 do
4:         if A[i] ≥ A[index_1] then
5:             index_1 = i
6:         end if
7:     end for
8:     index_2 ← 0
9:     for i ← 1, n − 1 do
10:        if i! = index_1&&A[i] ≥ A[index_2] then
11:            index_2 = i
12:        end if
13:    end for
14:    return A[index_1] ∗ A[index_2]
15: end procedure
```

Algorithm 8 Linear Search

```
1: procedure LINEAR(A, n, item)
2:     for i ← 0, n − 1 do
3:         if A[i] == item then
4:             return i
5:         end if
6:     end for
7:     return −1
8: end procedure
```

Algorithm 9 Bubble Sort

```
1: procedure BUBBLESORT(A, n)
2:     for k ← 0, n − 1 do
3:         for i ← 0, n − 1 do
4:             if A[i] > A[i + 1] then
5:                 swap(A[i], A[i + 1]
6:             end if
7:         end for
8:     end for
9: end procedure
```

Algorithm 10 Selection Sort

```
1: procedure SELECTIONSORT(A, n)
2:     for i ← 0, n − 1 do
3:         iMin ← i
4:         for j ← i + 1, n − 1 do
5:             if A[j] < A[iMin] then
6:                 iMin = j
7:             end if
8:             swap(A[iMin], A[i])
9:         end for
10:    end for
11: end procedure
```