COP 3502- Algorithm Analysis Exercise

Part A

1) For an $O(n^3)$ algorithm, one data set with n = 3 takes 54 seconds. How long will it take for a data set with n = 5?

2) For an $O(2^n)$ algorithm, a friend tells you that it took 17 seconds to run on her data set on a $O(2^n)$ algorithm. You run the same program, on the same machine, and your data set with n = 7 takes 68 seconds. What size was her data set?

3) For an $O(N^k)$ algorithm, where k is a positive integer, an instance of size M takes 32 seconds to run. Suppose you run an instance of size 2M and find that it takes 512 seconds to run. What is the value of k?

4) Assume that an $O(\log_2 N)$ algorithm runs for 10 milliseconds when the input size (N) is 32. What input size makes the algorithm run for 14 milliseconds?

5) An algorithm to process a query on an array of size n takes $O(\sqrt{n})$ time. For $n = 10^6$, the algorithm runs in 125 milliseconds. How many ***seconds*** should the algorithm take to run for an input size of n = 64,000,000?

7) An algorithm processing a two dimensional array with R rows and C columns runs in (RC^2) time. For an array with 100 rows and 200 columns, the algorithm processes the array in 120 ms. How long would it be expected for the algorithm to take when processing an array with 200 rows and 500 columns? Please express your answer in ***seconds. // for this question,the function should look like this:*** $T(R, C) = kRC^2$ ***where k is a constant***

8) A search algorithm on an array of size n runs in O(lg n) time. If 200,000 searches on an array of size 2^{18} takes 20 ms, how long will 540,000 searches take on an array of size 2^{20} take, in milliseconds?

Part B

The following questions, represented as functions with appropriate names, determine the run-time for the function in terms of the variable n. The answers should simply be Big-Oh answers, but you need to provide ample justification for your answers. You may assume that n is a positive integer.

Question 1

```
int function5(int A[], int B[], int n) {
        int i, j, sum = 0;
        for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            if (A[i] == B[j])
                sum++;
    return sum;
}
```

Question 2

```
int function6(int A[], int B[], int n) {
    int i=0,j=0;
    while (i < n) {
        while (j < n && A[i] > B[j]) j++;
        i++;
    }
    return j;
}
```

Question 3

```
int function7(int A[], int B[], int n) {
    int i=0,j;
    while (i < n) {
        j=0;
        while (j < n && A[i] > B[j]) j++;
        i++;
    }
    return j;
}
```

Question 4

```
void function8(int n) {
    while (n > 0) {
        printf(“%d\n”, n);
        n = n/2;
    }
}
```

Question 5

```
int function9(int n) {
    int i,j;
    for (i=0; i<n; i++)
        for (j=0; j<n; j++)
            if (j == 1)
                break;
    return j;
}
```

Question 6

Consider the following function with integer inputs n and m:

```
 void solveit(int* array, int n, int m)
 {
    int i, res = 0;
    for (i=0; i<n; i++)
    {
        int low = 0, high = m;
        while (low < high)
        {
            int mid = (low+high)/2;
            if (f(mid) < array[i])
                low = mid+1;
            else
               high = mid-1;
        }
        printf("%d\n", low);
    }
}
```

You may assume that the function f that is called from solveit defines a function that runs in O(1) time. With proof, determine the run-time of this function in terms of n and m.

Question 7:

Below is a program which includes a single function call to the function mysqrt. The function mysqrt includes a while loop. Give an estimate and analysis on how many times that while loop will run during the single function call from main.

```
int main() {
    printf(“%.6f\n”, mysqrt(1000));
    return 0;
}
double mysqrt(double n) {
    double low = 1, high = n;
    if (n < 1) {
          low = n;
          high = 1;
    }
    while (high - low > .000001) {
        double mid = (low+high)/2;
        if (mid*mid < n)
                 low = mid;
      else
             high = mid;
   }
}
```

Question 8:

What would be the worst case runtime for the push and pop function for a stack?

Part C Summation:

Before starting the summation exercises, go through the examples in the slides and try to solve them without looking at the solution if you think any example challenging to you.

Question 1

Determine the following summation in terms of n (assume n is a positive integer 2 or greater), expressing your answer in the form $an^3 + bn^2 + cn$, where a, b and c are rational numbers. (Hint: Try rewriting the summation into an equivalent form that generates less algebra when solving.)

$$\sum_{i=n^2-3}^{n^2+n-4} (\mathrm{i} + 4)$$

Question 2

1. There is a formula: $\sum_{i=0}^{n-1} 2^{\mathrm{i}} = 2^n - 1$

(a) Using this result, determine a closed-form solution in terms of *n*, for the summation below.

(b) Determine the numeric value of the summation for n = 9.

$$\sum_{i=0}^{n} \left(\sum_{j=0}^{i-1} 2^j \right)$$

Question 3

1. Let *a, b*, *c,* and *d*, be positive integer constants with *a* < *b*. Prove that

$$\sum_{i=a}^{b} (ci + d) = \frac{(c(a + b) + 2d)(b - a + 1)}{2}$$