CMPT 125, Fall 2021

Midterm Exam
October 25, 2021

Name_____

SFU ID: |__|__|__|__|__|__|__|__|__|

Problem 1	
Problem 2	
Problem 3	
Problem 4	
TOTAL	

Instructions:

1. Duration of the exam is 90 minutes.
2. Write your name and SFU ID **clearly**.
3. This is a closed book exam, no calculators, cell phones, or any other material.
4. The exam consists of four (4) problems. Each problem is worth 25 points.
5. Write your answers in the provided space.
6. There is an extra page at the end of the exam. You may use it if needed.
7. Explain all your answers.
8. Really, explain all your answers.

Good luck!

Problem 1 [25 points]

a) [6 points] What will be the output of the following program? Explain your answer.

```
#include <stdio.h>
enum numbers {ZERO, ONE, TWO, THREE};

void foo(int* x, int y) {
     int z = 5;
     *x = z;
     y = z;
     x = &y;
}

int main() {
     int a = ONE, b = TWO;
     foo(&a, b);
     printf("a = %d, b = %d", a, b);
     return 0;
}
```

b) [6 points] Will the code below compile?
If yes, what will be the output? If not, explain why.

```
#include <stdio.h>

void str_manipulate(char* s) {
    char c = '\0';
    *(s+2) = c;
}

int main() {
    char str[13] = "CMPT125";
    str_manipulate(str+3);
    printf("%s\n", str+1);
    return 0;
}
```

c) [7 points] Will the code below compile?
If yes, what will be the output? If not, explain why.

```
#include <stdio.h>

int* get_arr3() {
  int arr[5] = {1,2,3};

  int* ret = arr;
  return ret;
}

int main() {
  int* a1 = get_arr3();
  int* a2 = get_arr3();
  a1[0] = 7;
  a2[1] = 8;
  printf("a1 = [%d, %d, %d]\n", a1[0], a1[1], a1[2]);
  printf("a2 = [%d, %d, %d]\n", a2[0], a2[1], a2[2]);
  return 0;
}
```

d) [6 points] What is the running time of the function below. Use Big-O notation to express your answer. Explain your solution.

```
int foo(int n) {
  if (n>0) {
    int sum = 0;
    for(int i=0; sum<n ; i++) // note the condition is sum < n
      sum = sum+i;
  }
  return i+foo(n-1);
}
```

Problem 2 [25 points]

a) [5 points] Consider the **Binary Search** algorithm. How many comparisons will it make on the input A = [2, 4, 6, 8, 10, 12, 14, 15, 18, 90, 100] when searching for 7. Explain your answer

b) [8 points] Give an example of an array of length n, on which **InsertionSort** makes exactly one swap in each iteration of the outer loop.

c) [6 points] Recall the **MergeSort** algorithm.

```
void merge_sort(int* A, int n) {
  if (n >= 2) {
    int mid = n/2;
    merge_sort(A, mid);          // sort left half
    merge_sort(A+mid, n-mid);    // sort right half
    merge(A,n,mid); // merge() we saw in class with running time O(n)
  }
}
```

What is the running time of this algorithm when applied on a sorted array of length n?
Write the tightest possible upper bound on the running time. Explain your answer.

d) [6 points] Consider the following variant of **MergeSort**:

```
bool is_sorted(int* A, int n); //checks if A is sorted in O(n) time

void merge_sort(int* A, int n) {
  if (n <= 1 || is_sorted(A, n)) // stopping condition
    return;

  int mid = n/2;
  merge_sort(A, mid);          // sort left half
  merge_sort(A+mid, n-mid);    // sort right half
  merge(A,n,mid); // merge() we saw in class with running time O(n)
}
```

What is the running time of this algorithm when applied on A=[n, n-1, n-2, n-3, n-4,… 3, 2, 1]?
Write the tightest possible upper bound on the running time. Explain your answer.

Problem 3 [25 points]

a) [8 points] Write a function that gets two strings and computes the length of their longest common prefix. For example,

- longest_common_prefix(“abcd”, “ab12”) is 2 - the prefix is “ab”
- longest_common_prefix(“abcd”, “cd”) is 0 - the prefix is empty
- longest_common_prefix(“abcd”, “abcdefg”) is 4 - the prefix is “abcd”

Explain your idea before writing code.

```
int longest_common_prefix(const char* str1, const char* str2) {









}
```

[4 points] What is the running time of your function? Use big-O notation to state your answer. Give the tightest possible answer.

b) [10 points] Implement the function strcat. The function gets *dest* and *src*, and appends the string pointed to by *src* to the end of the string pointed to by *dest*.
This function returns a pointer to the resulting string dest.

For example, if dest = “ABC” and src = “DEF”, then after the function returns we have dest = “ABCDEF”.

You are not allowed to use any library functions to solve this.

```
// assumption: dest has enough memory allocated
// to store the concatenation of the two strings
char* strcat(char* dest, const char* src) {









}
```

[3 points] What is the running time of your function in terms of the length of the strings in the worst case? Use big-O notation to state your answer. Give the tightest possible answer.

Problem 4 [25 points]

a) [8 points] Write a function that gets a parameter n. It reads n ints from the user (using scanf), and prints the numbers in the reverse order.

```
void read_and_print_reverse(int n) {




}
```

b) [15 points] Write a function that gets an array of ints of length n, and returns an array of length n, such that the output[i] is equal to the maximal element in the input subarray [0,..., i]. For example,

- input [1, 4, 3, 8, 2, 9]
- output [1, 4, 4, 8, 8, 9].

You may write helper functions if that makes the solution more readable.

- A correct answer with linear running time, will give you 15 points
- A correct answer with quadratic running time, will give you 10 points

```
int* max_prefixes(int* ar, int n) {









}
```

[3 points] Explain the running time of your function.

Extra page

Empty page