CMPT 125 D100, Spring 2022

Midterm Exam
March 4, 2022

Name_____

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

Problem 1	
Problem 2	
Problem 3	
Problem 4	
TOTAL	

Instructions:

1. Duration of the exam is 100 minutes.
2. Write your full 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 colors {RED, GREEN, BLUE, YELLOW};

void foo(int* x, int* y) {
     long z = 3;
     *y = 6;
     x = y;
     *x = z;
}

int main() {
     int a = GREEN, b = YELLOW;
     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>

int main() {
    char s[10] = {'A','B',0,'C','D','E','F','G',0};
    int what=0;
    while (str[what]) {
      what = what+1;
    }
    printf("%d\n", what);
    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* bar_arr() {
  int arr[4];
  for(int i=0;i<4;i++)
    arr[i]=1;
  int* ret = arr;
  return ret;
}

int main() {
  int* a1 = bar_arr();
  a1[0] = 2;
  printf("a1 = [%d, %d, %d, %d]\n", a1[0], a1[1], a1[2], a1[3]);
  return 0;
}
```

d) [6 points] Let T(n) be the running time of foo(1,n). Use Big-O notation to express T(n). Explain your solution.

```
void foo(int k, int n) {
  if (k<=n) {
    int mid = (k+n)/2; // if (k+n) is odd, (k+n)/2 is rounded down
    for(int i=k; i<=mid ; i++)
      printf("i = %d  ", i);
    printf("\n");
    foo(mid+1, n);
  }
}
```

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, 9, 12, 14, 15, 18, 90, 99] when searching for 15? Explain your answer.

b) [8 points] Show an array with the values {1, 2, 3, 4, 5, 6, 7, 8} so that the **InsertionSort** makes exactly 6 swaps in the last iteration of the outer loop, and makes no other swaps.

c) [12 points] Implement the merge function that gets an array A of length n, and index mid, such that A[0,...mid] and A[mid+1…n-1] are sorted in the increasing order.
The function merges the two halves of A into a sorted array in time O(n).
* Note that some elements might be equal.
Remember to use malloc/free if you need to use a new array.

Explain your code if necessary.

```
void merge(int* A, int n, int mid) {









}
```

Problem 3 [25 points]

a) [8 points] Write a function that gets a string and computes the length of its longest prefix consisting only of the lowercase letters. For example,

- longest_lower_case_prefix("abCDef") is 2 - the prefix is "ab"
- longest_lower_case_prefix("12abcd") is 0 - the string starts with "12"
- longest_lower_case_prefix("abc") is 3 - the prefix is "abc"

Explain your idea before writing code.

```
int longest_lower_case_prefix(const char* str) {









}
```

[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 a function that gets a string *str*, and a positive integer k, and searches for a substring of str of length exactly k that is a palindrome.
The function returns the index in *str* representing the beginning of such a palindrome.
If *str* contains more than one such palindrome, the function returns the index to the first palindrome of length k.
If *str* does not contain a palindrome of length k, the function return -1
For example,

- find_k_palindrome("**ABBA**bcd", 4) returns 0.
- find_k_palindrome("Hell**owo**rld wowowow", 3) returns 4.
- find_k_palindrome("**wow**owowHelloworld", 3) returns 0.
- find_k_palindrome("D**ABCDCBA**", 7) returns 1.
- find_k_palindrome("Rolling Stones", 6) returns -1.

You are not allowed to use any library functions to solve this, except for strlen().

```
char* find_k_palindrome(const char* str, int k) {




}
```

[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]

Consider the following function.

```
int what(unsigned int n) {
  if (n<=2)
    return n;
  return (n-1)*what(n-1) + (n-2)*what(n-2);
}
```

a) [4 points] Compute what(4). Explain your answer.

b) [9 points] Rewrite the function what() with the same functionality so that on input n, it returns the answer in time O(n). Explain your answer.

c) [12 points] Write a function that gets an array of ints of length n, and returns an array of length n, such that output[i] is equal to the maximal element in the input subarray [i…n-1]. For example,

- input [1, 4, 9, 8, 2, 5]
- output [9, 9, 9, 8, 5, 5].

Make sure the returned array is allocated on the heap.

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_suffix(const int* ar, int n) {









}
```

Extra page

Empty page