APS 105H1 S 2010 Final Exam
Duration — 150 minutes
Aids allowed: none

Student Number: |_|_|_|_|_|_|_|_|_|

Last Name: ____________________ **First Name:** ____________________

Lecture Section: 1 **Instructor: Daniel Zingaro**

Do **not** *turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the exam**, and read the instructions below.)
Good Luck!

This exam consists of 9 questions on 16 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* If you use any space for rough work, indicate clearly what you want marked.

# 1:	______/17
# 2:	______/ 5
# 3:	______/ 6
# 4:	______/ 6
# 5:	______/ 6
# 6:	______/ 6
# 7:	______/ 7
# 8:	______/12
# 9:	______/ 7
TOTAL:	______/72

Question 1. [17 MARKS]

Part (a) [3 MARKS]

Consider the following code.

```
char *s = "";
if (s == NULL)
  printf ("inside if\n");
```

Carefully explain when the **printf** in the code above would execute.

Part (b) [3 MARKS]

Write a program fragment to print the elements of a two-dimensional array a from top to bottom, right to left. For example, for the following two-dimensional array:

2	4	6
12	14	16
1	2	3

We will print `6 4 2 16 14 12 3 2 1`. Assume that a has been declared as `int a[ROWS][COLS]`, and that the elements of a have all been initialized.

Part (c) [3 MARKS]

We discussed two techniques that can be used to write functions that can give back multiple values to their callers: (1) using structures, and (2) using pointers. Briefly describe how each technique allows a function to communicate more than one value back to its caller.

Part (d) [2 MARKS]

When we pass an array to a function, we typically also pass the array's length as a separate parameter. What is the reason for passing the array's length?

Part (e) [3 MARKS]

The following recursive function expects a positive integer parameter n, and prints the integers from n down to 1. For example, if we call the function with parameter 3, it prints 3 2 1. Fill in the correct code for the base case.

```
void countdown (int n) {
  if (n == 0) {
    //base case code goes here




  }
  printf ("%d\n", n);
  countdown (n-1);
}
```

Part (f) [3 MARKS]

Assume that we want to repeatedly ask the user for a character until they provide:

- One of the characters 1, 2, ..., 9, or
- The lowercase character q

Once the user has provided a character that meets one of these two conditions, we want to store it in char variable ch. Give the while-loop condition that we would use. Assume that ch has already been declared.

```
while (
```

Question 2. [5 MARKS]

A certain lottery uses tickets that each contain three positive integers. Call these integers `a`, `b`, and `c`. For each ticket you purchase, we can determine the amount of money you will win by the following rules:

- (1) If `a`, `b`, or `c` is 0, you win 0 dollars
- (2) Otherwise, you win `a + b + c` dollars
- (3) As a special bonus, if `c` is larger than the other two numbers, you win twice as much as what rule (2) says

Write a function **`winnings`** that returns the amount you win on a ticket, using the above rules. The following table gives several examples of what the function should return.

Function call	Returns
`winnings (2, 6, 3)`	11
`winnings (2, 3, 6)`	22
`winnings (2, 3, 0)`	0

```
int winnings (int a, int b, int c)
```

Question 3. [6 MARKS]

Write a function according to the specification below.

```
void odd_even (int n, int a[n])
```

Rearrange elements of `a` so that all of its odd integers are followed by all of its even integers. The final order of the odd integers, and the final order of the even integers, does not matter (but make sure that all odd integers come first). For example, one correct execution of the function on the array
`2 7 3 4 5`
is to rearrange the array as
`7 3 5 2 4`.

Question 4. [6 MARKS]

We would like to write our own version of C's built-in `strcat` function, without using any of C's other string-processing functions.

On the next page, you are given 11 fragments of code in random order. Your task is to use each of these fragments **at least once**, and add the proper opening and closing braces (`{ }`), to arrive at a function that does the same thing as `strcat`. Do **not** add new code; simply copy and rearrange the code from the fragments.

- `p++;`
- `i++;`
- `while (*p != '\0')`
- `int i = 0;`
- `*p = s2[i];`
- `p = s1;`
- `while (s2[i] != '\0')`
- `char *my_strcat (char *s1, const char *s2)`
- `*p = '\0';`
- `char *p;`
- `return s1;`

Question 5. [6 MARKS]

Write a function according to the specification below.

```
bool block_string (int n, char *s, char c)
```

`s` is a string (i.e. a `char` array), `n` is the length of `s`, and `c` is a character. Return `true` exactly when `s` consists entirely of blocks of character `c`, each of which is one character longer than the previous block. For example,
`block_string ("a aa aaa ", 'a')`
returns `true`, but
`block_string ("aa a aaa ", 'a')`
returns `false`. Note that each block of characters, **including the last**, must be followed by a space.

Question 6. [6 MARKS]

What is the output we get from running the following C program? Please show your intermediate work as you trace through the code; this helps us award part marks and helps you keep track of values of variables and actions of recursive calls.

```
#include <stdio.h>

void recur (int a, int b) {
  if (a + b < 0)
    return;
  recur (a - 1, b - 2);
  printf ("%d %d\n", a, b);
}

int main (void) {
  recur (6, 0);
  return 0;
}
```

Question 7. [7 MARKS]

Two characters in a grid are connected if we can get from the first character to the second by a sequence of zero or more horizontal or vertical steps, each of which places us on a non-empty-space character. For example, consider the following grid:

```
XYQ
G B
 N
P
```

As examples:

- X is connected to X
- Y is connected to B
- N is not connected to any other character

Write a function that takes a two-dimensional array of characters, and the x-and y-coordinates of two characters. The function returns **true** exactly when the second character is reachable from the first. Each of the two-dimensional array's values will be one of the 26 uppercase letters of the alphabet, or the space (' ') character. The space character is the only character that cannot be used to connect two characters. Assume that the borders of the two-dimensional array are all space characters.

```
bool connected (char grid[][COLS], int x1, int y1, int x2, int y2) {
```

Question 8. [12 MARKS]

For the following questions, all sorting algorithms are assumed to be the versions studied in class.

Part (a) [3 MARKS]

What are the contents of the following array of numbers after one major iteration of bubble sort? Be sure to use one major iteration of a bubble sort that sorts from lowest to highest, and show the array after each change you make.

```
5 4 3 8 3 3
```

Part (b) [2 MARKS]

After k major iterations of a sorting algorithm ($k \geq 1$), our array of numbers looks like this.

```
1 3 10 1 2 3
```

Is it possible that selection sort is being used to sort this array? Answer **yes** or **no**, and then briefly justify your answer.

Part (c) [2 MARKS]

After k major iterations of a sorting algorithm ($k \geq 1$), our array of numbers looks like this.

```
1 3 10 1 2 3
```

Is it possible that insertion sort is being used to sort this array? Answer **yes** or **no**, and then briefly justify your answer.

Part (d) [2 MARKS]

Which sorting algorithm, selection sort or insertion sort, would perform fewer comparisons on an already-sorted array? Briefly justify your answer.

Part (e) [3 MARKS]

Consider the following array:
`4 1 6 3 9 3`

What are the contents of the array after using quicksort's partition algorithm with pivot `3`? Please show your work. (Perform only one partition of the array; do **not** continue to recursively sort the two resulting subarrays.)

Question 9. [7 MARKS]

Part (a) [3 MARKS]

We have 15 students enrolled in a lecture session, and we would like to store each of their final grades in order to calculate the class average. Each grade is an integer between 0 and 100. What should we use to store the 15 student grades: (1) a structure, or (2) an array? Briefly justify your answer.

Part (b) [4 MARKS]

Here is a `struct` used for holding the three components of a date.

```
struct date {
  int month, day, year;
}
```

Write a function that takes three `int` parameters representing the `month`, `day`, and `year` of a date. The function should allocate a new `date` structure, fill it with the values passed in the three parameters, and then return a pointer to the structure.

```
struct date *make_date (int month, int day, int year)
```

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]