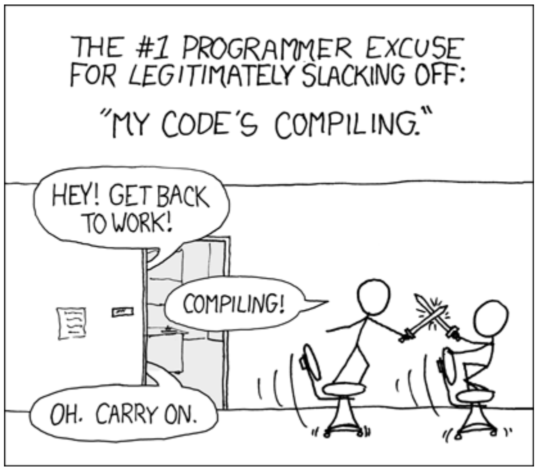
First Name: ________________________

Autumn 2017 Last Name: ________________________

Exam for CSE 220 (2017)

Answer the questions in the spaces provided on the page. If you run out of room for an answer, continue on the back of the page.

- DO NOT OPEN THE EXAM UNTIL TOLD TO DO SO
- You only need to answer 6 of the 7 questions.
- On one of the questions, make a large slash across the page, which indicates that it should not be graded.
- On every page (including the first and last page), write your first and last name, before answering the question. Unnamed pages may be lost. And points may be deducted.
- If you start to answer a question and then change your mind, please cross out the attempt and write *DO NOT GRADE* across it.
- Legibility matters! If we can't read your answer, you will receive a 0 for it.



https://xkcd.com/303/

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 1: Reading from Standard In . . . . *50 points*

I need a program that can read in some input and store the information into some variables. My input is formatted as follows:

```
RaceTrack - 4 "ferret" years old, M
```

In the above, the name is "RaceTrack", the age is "4", and the sex is "M". Write a line that will assign to these variables by reading from stdin. I know that no possible name is more than 100 characters. Be sure that you don't allow names longer than 100 characters to crash your program.

Here is my code:

```
#include <stdio.h>
#define NAME_MAX_SIZE 100
int main(void) {
    int age;
    char sex;
    char name[NAME_MAX_SIZE + 1];
    // YOUR LINE HERE
    return 0;
}
```

What should `//YOUR LINE HERE` be replaced with?

```
scanf("%100s - %d \"ferret\" years old, %c", name, &age, &sex);
```

Points earned: __________ out of a possible 50 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Grade Breakdown

- Having scanf - 10 pts
- Take in 3 variables with correct types and order: 15 pts (doesn't assign input to 3 variables - 7 pts; correct types - 6 points; order of variables - 2 points)
- Having "\ferret\" years old - 10 points (backslashes - 3 points; "ferret" years old - 7 pts)
- Syntax - 15 points (Semicolon - 5 points; Ampersands, %, semicolon = 10 pts)

Points earned: ____________ out of a possible 0 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 2: Repeat String .... *50 points*

Write a function named ”repeat”, that takes a char * (string) as its only argument. This function should repeat the contents of the string, making the number of non-null characters double. For example, if the input string is ”cat”, after calling repeat, the string should be ”catcat”. You may not make any function calls in this function. You may assume the char array holding the string has enough room for the resulting string.

Solution:

```
void repeat(char * str) {
    char * old_end = str;
    while (*old_end != '\0') {
     ++old_end;
    }
    char * ptr = old_end;
    while(str != old_end) {
        *ptr = *str;
        ++ptr;
        ++str;
    }
    *ptr = '\0';
}
```

Points earned: ____________ out of a possible 50 points

First Name: ______________________

Autumn 2017 Last Name: ______________________

Grade Breakdown

- correct return type +5
- correct function name and correct argument +5 (+2 if only correct function name, +3 for correct argument)
- using a loop to find the length of the string/iterating through string until reaching null character +10 (+8 if this was attempted but had small error, +5 if attempted and did not complete or had major error)
- adding a null character to the end of the string after repeating its contents +10 (+8 if this was attempted but had small error)
- formatting/syntax/other +5 had to have something in the function body to earn any points here (+2 - +3 if syntax was good but little code was written/program was incomplete)
- doubling the string +10 (+5 for using a loop to repeat the string contents, +5 for correct logic in loop)
- correct final result +5 (+2 - +3 if only a small flaw in the previous logic prevented final string from being correct)

Points earned: ___________ out of a possible 0 points

First Name: ______________________

Autumn 2017 Last Name: ______________________

Question 3: Input and Output .... *50 points*

Below is a program which uses a recursive functions. For each of the supplied inputs, write what the program would output. If the program would perform an illegal action, write "illegal" instead of the output.

```
#include <stdio.h>
void recur(char * in);
void abc(char *in) {
    *in = '1';
    recur(in + 1);
}
void def(char *in) {
    *in = '0';
    recur(in + 2);
}
void recur(char * in) {
    if (*in == '\0') {
        return;
    }
    if ((*in >= 'A') && (*in <= 'Z')) {
        abc(in);
    } else {
        def(in);
    }
}
int main(void) {
    char str[6];
    scanf("%s", str);
    recur(str);
    printf("%s", str);
}
```

(a) (5 points) `B` ____`1`____

(b) (10 points) `aBB` ____`0B1`____

(c) (10 points) `aaBBBa` ____`illegal (dereferenced past end of array)`

(d) (15 points) `if1YK` ____`0f0Y1`____

(e) (10 points) `!=YES` ____`0=111`____

Points earned: __________ out of a possible 50 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 4: No Indexing . . . . *50 points*

I want a function (”strip”) that helps me strip leading whitespace (spaces and tabs) from my strings. For example, the string ”\t hello” has two leading whitespace characters (a tab and a space). I'm also interested in the number of leading whitespace characters there were. Write a function that takes one string argument (the string with possible leading whitespace) and a pointer to an int (where the number of leading whitespace characters should be stored). This function should return a pointer to the first non-whitespace character.

You need to write the function (named ”strip”), but **you are not allowed to use the characters `[` or `]`**.

```
// Example use:
char * string = "\t \t josh is cool \t ";
int num_ws;
char * a = strip(string, &num_ws);
printf("%s %d", a, num_ws); // Should print "josh is cool      4"
```

Solution:

```
char * strip(char * str, int * num_ws) {
    *num_ws = 0;
    for (; *str != '\0'; ++str) {
        if (*str != ' ' && *str != '\t') {
            return str;
        }
        ++(*num_ws);
    }
    return str;
}
```

Points earned: ____________ out of a possible 50 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Grade Breakdown

- Correct return type: +2
- Function name: +3
- Function arguments with correct types: +10 (+5 each)
- Function arguments with correct types: +10 (+5 each)
- Initialize num_ws: +5 (+3 if attempted a counter for whitespace)
- Loop through the string: +10 (+8 if completed with minor errors; +5 if completed with major errors or attempted)
- If statement to check for leading spaces: +10 (+5 if ' ' is checked, +5 if '/t' is checked; +3 if only attempted)
- Correctly returning string: +5 (+3 if attempted)
- Incrementing num_ws: +5 (+4 if attempted with minor errors; +2 if attempted with major errors)

Points earned: ____________ out of a possible 0 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 5: Pointers and Arrays . . . . *50 points*

For each code section, write what would be outputted by the code. If the code performs illegal actions, write "illegal".

(a) (10 points)
```
int j = 'c'; int k = 3;
char str[] = {'A', 'B', 'C', 'D', 'E'};
str[k] = j;
printf("%s", str);
```

`illegal (str is doesn't have a null character)`

(b) (10 points)
```
int j = 10; int k = 15;
int * m = &j; k = j; *m = 4;
printf("%d:%d:%d", j, k, *m);
```

`4:10:4`

(c) (10 points)
```
char a[] = "1234";
char *b = a; b += 2; *b = '9';
printf("%s %s", a, b);
```

`1294 94`

(d) (10 points)
```
char a[] = "the end"; char *b = a; int i = 0;
for ( ; b[i] != ' '; ++i) {
    printf("%d--", i);
}
printf("(%d)", (&b[i]) - a);
```

`0--1--2--(3)`

(e) (10 points)
```
char a[] = "abcd"; char * b = "xyz";
for (int i = 0; b[i] != '\0'; ++i) {
    *(a + i) = b[i] - 1;
}
printf("%s %s", a, b);
```

`wxyd xyz`

Points earned: ____________ out of a possible 50 points

First Name: ______________________

Autumn 2017 Last Name: ______________________

Grade Breakdown

- +3 if attempted output when the code wasn't illegal
- +5 if partially correct output
- +10 full points

Points earned: ___________ out of a possible 0 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 6: Lengths of Arrays . . . . *50 points*

What are the size of the array named 's' for the following array initializations? If a statement is illegal, write "illegal".

(a) (5 points) `char s[] = "";` ____**1**____

(b) (5 points) `char s[3] = "a";` ____**3**____

(c) (5 points) `char s[5]; s[2] = '\0';` ____**5**____

(d) (5 points) `char s[] = "abd\n\t\n";` ____**7**____

(e) (5 points) `char s[] = "'c'";` ____**4**____

(f) (5 points) `char s[] = {'a', 'b', 'd'};` ____**3**____

(g) (5 points) `char s[10] = {'a', 'b', 'd'};` ____**10**____

(h) (5 points) `char s[4] = "hello";` ____**illegal**____

(i) (5 points) `char s[] = "abc" "de";` ____**6**____

(j) (5 points) `char s[4] = "123" + "45";` ____**illegal**____

Points earned: __________ out of a possible 50 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Question 7: Using Pointers and Functions . . . . *50 points*

I have written an incredible function, called ”favorites”, that when given a person’s birth date, can ”scientifically” determine what that persons favorite color (string with less than 10 non-null characters), favorite number (int), and favorite letter (char) should be. Here is the function’s declaration:

```
void favorites(int birth_year, int birth_month, int birth_day,
   char * fav_color, int * fav_number, char * fav_letter);
```

Write a main that uses the function ”favorites” to determine my favorites. My birth day is 1, birth month is 11, and birth year is 1988. Print (to standard out) my favorite color, number, and letter. The specific output format is up to you.

Solution:

```
int main(void) {
char color[10];
    int num;
    char letter;
    favorites(1988, 11, 1, color, &num, &letter);
    printf("%s %d %c", color, num, letter);
    return 0;
}
```

Points earned: ____________ out of a possible 50 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

Grade Breakdown

- +10 for proper variable types and initializations (-5 for using variables without declaring them; -3 for incorrect type)
- +10 for printing the output variables (-1 per incorrect parameter; -10 if no print at all)
- +20 for calling favorites (-3 per incorrect parameter; -20 if favorites not called; -10 if parameter are in the wrong order)
- +10 for syntax (-2 for few syntax error; -4 for a moderate amount of syntax errors; -6 for many syntax errors)

Points earned: ____________ out of a possible 0 points

First Name: ________________________

Autumn 2017 Last Name: ________________________

If you have finished early, feel free to bring your exam to an instructor.
Or, you can draw a picture of your favorite Pokémon.
Or, you can write a haiku about your love of multidimensional arrays.

Question	Points	Score
Reading from Standard In	50	
Repeat String	50	
Input and Output	50	
No Indexing	50	
Pointers and Arrays	50	
Lengths of Arrays	50	
Using Pointers and Functions	50	
Total:	300	

We tallied the top 4 highest scoring questions, then multiplied by 1.5 to get your score out of 300 points.