CS 2110 — Quiz 5, Group A
March 2018

Full name: ____________________
GT username: ____________________
This quiz is worth a total of **100 points**.

Pointers and Operators

1. For each line in the following table, show the updated value of the variable after the line is executed. You must have exactly one entry in each row. Use the & operator to denote the address of a variable. $\boxed{21}$

Code	d	a	n	pa	pd	ppa	ppd
int d = 15;	15						
int a = 32;		32					
int n = 16;			16				
int *pd = &d;					&d		
int **ppd = &pd;							&pd
int *pa = &a;				&a			
int **ppa = &pa;						&pa	
*pd = *pa;	32						
*ppd = pa;					&a		
*pd = n;		16					
pd = &a;					&a		
pa = &n;				&n			
**ppd = d;		32					
**ppa = a;			32				

Multiple Return Values

2. Below is a file containing a function called *smartDivision*, as well as a main function which makes a valid call to *smartDivision*. *smartDivision* should divide a given integer *a* by a given integer *b*, return the quotient and the remainder of $a \div b$ through its parameters *q* and *r* respectively, and return an integer representing the divisibility of *a* by *b* as its return value. It should return 1 if *a* is divisible by *b* and 0 otherwise. *Hint: Checking for divisibility is equivalent to checking if the remainder is zero.*

   (a) Complete the function header with the correct return type and the correct types for the parameters. $\boxed{10}$

   (b) Then complete the function body to achieve the results explained above. $\boxed{10}$

```
__int___ smartDivision(__int___ a, __int___ b, __int*___ q, __int*___ r) {


        *q = a/b;
        *r = a%b;

        return *r == 0;


}

int main() {
    int quotient, remainder;
    int divisible = smartDivision(10, 3, &quotient, &remainder);
    \\ ... rest of main function uses divisible, quotient and remainder
}
```

5pts for correct #define and arguments, 2pts for correct implementation (sum or product)
4pts for argument parens, 4pts for surrounding parens,
-5pts for syntax errors not covered above (like semicolon)

Defining a Macro

3. Define a macro that takes three parameters a, b and c calculates the product (multiplication) of the three parameters. 15

```
#define MULT(a, b, c) ((a) * (b) * (c))
```

Pointers, Arrays and Structs

4. Fill in the blanks in the question in accordance with the explanations in the comments. 24

```
typedef struct { int x, y; } CoordinatePair;
CoordinatePair *pairs[10];
// Assume "pairs" is filled with valid (non-NULL) CoordinatePair pointers

// Fill in the blanks so that when it is executed, the variable "ptr" has
// the 3rd pointer in the array. Hint: the first blank is related to the
// variable's type.

_CoordinatePair____ *ptr = _____pairs[2]______; // 8 points

// Fill in the blanks so that when it is executed, the variable "pair" has
// the CoordinatePair pointed to by the 5th pointer in the array. Hint: the
// first blank is related to the variable's type.

_CoordinatePair____ pair = ____*pairs[4]_____; // 8 points

// Fill in the blank so that the "x" field of the CoordinatePair pointed to
// by the 4th pointer in the array is set to equal the "y" field of the
// CoordinatePair pointed to by the 6th pointer in the array. Hint: both
// blanks are expressions.

_pairs[3]->x____________ = __pairs[5]->y___________; // 8 points
```

Storage Classes and Type Qualifiers

5. For each of the contexts given below, explain what effect the keyword has.

(a) *static* as a modifier on a variable definition inside a function. 5

Makes variable value persist across invocations. It is thus stored in static memory and not on the stack.

(b) *static* as a modifier on a variable definition outside a function. 5

Makes variable invisible from other files.

(c) *static* as a modifier on a function definition. 5

Makes function invisible from other files i.e. it can only be called from within the same file.

(d) *volatile* as a modifier on a variable definition. 5

Makes compiler not optimize accesses to this variable (i.e. repeated reads from memory will not be replaced with caching in register, useful for cases where the value may change externally)

NOT ON CS2110 FALL 2018 QUIZ 5