

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.

21

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

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.

10

(b) Then complete the function body to achieve the results explained above.

10

```

1  _____ smartDivision(_____ a, _____ b, _____ q, _____ r) {
2
3
4
5
6
7
8
9  }
10
11 int main() {
12     int quotient, remainder;
13     int divisible = smartDivision(10, 3, &quot;quot;quotient, &remainder);
14     \\ ... rest of main function uses divisible, quotient and remainder
15 }
```

Defining a Macro

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

15

Pointers, Arrays and Structs

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

24

```
1 typedef struct { int x, y; } CoordinatePair;
2 CoordinatePair *pairs[10];
3 // Assume "pairs" is filled with valid (non-NULL) CoordinatePair pointers
4
5 // Fill in the blanks so that when it is executed, the variable "ptr" has
6 // the 3rd pointer in the array. Hint: the first blank is related to the
7 // variable's type.
8
9 ----- *ptr = -----; // 8 points
10
11 // Fill in the blanks so that when it is executed, the variable "pair" has
12 // the CoordinatePair pointed to by the 5th pointer in the array. Hint: the
13 // first blank is related to the variable's type.
14
15 ----- pair = -----; // 8 points
16
17 // Fill in the blank so that the "x" field of the CoordinatePair pointed to
18 // by the 4th pointer in the array is set to equal the "y" field of the
19 // CoordinatePair pointed to by the 6th pointer in the array. Hint: both
20 // blanks are expressions.
21
22 ----- = -----; // 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

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

5

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

5

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

5

NOT ON CS2110 FALL 2018 QUIZ 5