CSE 351 Study Guide 2 – Task 3

Last Name:	
First Name:	
UW NetID (username):	
Academic Integrity Statement: All work on these questions is my own. I had no prior knowledge of the questions, and I will not share or discuss my answers with anyone else. Violation of these terms may result in a failing grade. (**please sign**)	

Instructions

- Fill in your name and UW NetID in the table above, then read the Academic Integrity Statement and sign your name in the box to the right of it indicating that you understand and will comply with the statement.
- If able, fill in your responses on a copy of this document and upload your submission to Gradescope when completed.
- When uploading to Gradescope, mark which pages correspond to each question.
- Show scratch work for partial credit but answer in the blanks, boxes, or spaces provided.
- You may use your study guide from Task 1, course lecture slides and Ed Lessons, and course textbooks while completing this task.
- Use of reference materials external to those listed above is not allowed (e.g., Stack Overflow, web searches, etc.)
- These questions should take approximately 30 minutes to answer.
- Refer to the Study Guides webpage for additional information: https://courses.cs.washington.edu/courses/cse351/21wi/guides/

Advice

- Read each question carefully.
- Relax and breathe; you are here to learn.

Part 1. x86-64, The Stack, Procedures, Executables, Buffer Overflow

(A) The ISA specifies the names of all of the architecture's registers but does not specify their size. Circle your answer.

True / False

(B) Assume procedure P calls procedure Q and P stores a value in register `%rdx` prior to calling Q. After Q returns control to P, can P safely use the register `%rdx`? Circle your answer.

True / False

(C) Assuming normal execution (i.e., no segmentation faults or other errors occur), a procedure Q that is called by a procedure P will always return before procedure P returns. Circle your answer.

True / False

(D) Explain why they use of the gets() function introduces a security vulnerability when used in a program. Respond in one or two sentences.

(E) Consider the following definition of a struct in C.

```
struct particle {
  short x;
  short y;
  short z;
  double mass;
  unsigned short id;
};
```

a. How many bytes does an instance of this struct use? How many of those bytes are *internal fragmentation* and *external fragmentation*?

Total Bytes	Internal Fragmentation	External Fragmentation

b. True or False? It is possible to re-order the fields of this struct to reduce the total size of the struct. Circle your answer.

True / False

(F) Consider the following recursive function foo():

```
0000000000001139 <foo>:
    1139: b8 00 00 00 00        mov    $0x0,%eax
    113e: 85 ff                 test   %edi,%edi
    1140: 75 01                 jne    <foo+0xa>
    1142: c3                    retq
    1143: 53                    push   %rbx
    1144: 89 fb                 mov    %edi,%ebx
    1146: d1 ef                 shr    $0x2, %edi
    1148: e8 ec ff ff ff        callq  <foo>
    114d: 83 e3 01              and    $0x1,%ebx
    1150: 01 d8                 add    %ebx,%eax
    1152: 5b                    pop    %rbx
    1153: c3                    retq
```

a. How much space (in bytes) does this function take up in our final executable?

b. What callee-saved registers (if any) are used? Answer with the 64-bit register names.

c. What caller-saved registers (if any) are used? Answer with the 64-bit register names.

d. In one or two sentences, explain what this function does?

e. What is the return address to foo() that gets stored on the stack during the recursive calls? (provide your answer in hex)

Part 3. Caches

Assume we are executing on a system with the following parameters.

- Addresses are 16 bits wide
- There is a Level 1 Data Cache (L1 D$) that is four-way set associative with 32-byte blocks and 8 sets.
- The L1 D$ uses the write-back and write-allocate policies.
- The L1 D$ has a Hit Time of 3 ns, Hit Rate of 95%, and Miss Penalty of 20 ns
- The processor takes 1 ns to complete each clock cycle

(A) Fill in the number of bits allocated to each part of the address TIO breakdown:

Tag	Index	Offset

(B) How many Management Bits are required for each cache line?

(C) Assuming the other parameters remain unchanged, increasing the Associativity of the cache decreases Compulsory misses? Circle your answer.

True / False

(D) Assuming the other parameters remain unchanged, increasing the number of Sets decreases Capacity misses? Circle your answer.

True / False

(E) Compute the Average Memory Access Time (AMAT), in nanoseconds, for this cache.

(F) Determine the miss rate of the following C code, assuming it executes on the system described above. Only consider accesses to the arrays x and y. Recall that Miss Rate = (total misses / total accesses). Assume x is at address 0 and y is at address 128.

```
int foo(int x[32], long y[16])
{
  int sum = 0;
  for (int i = 0; i < 16; i++)
  {
    sum += y[i] + x[(i*2)] + x[(i*2)+1];
  }
  return sum;
}
```