CSE 351 Section 8 – Additional Problems from Autumn 2017 Final

1. Caching

We have 64 KiB of RAM and a 2-KiB L1 data cache that is 4-way set associative with 32-byte blocks and random replacement, write-back, and write allocate policies.

(A) Calculate the TIO address breakdown: [1.5 pt]

Tag bits	Index bits	Offset bits

(B) How many management bits (bits *other* than the block data) are there in every line in the cache? [1 pt]

_____ bits

(C) The code snippet below accesses an array of `doubles`. Assume `i` is stored in a register. Calculate the **Miss Rate** if the cache starts *cold*. [2.5 pt]

```
#define ARRAY_SIZE 256
double data[ARRAY_SIZE];        // &data = 0x1000 (physical addr)
for (i = 0; i < ARRAY_SIZE; i += 1)
    data[i] /= 100;
```

(D) For each of the proposed (independent) changes, write **IN** for "increased", **NC** for "no change", or **DE** for "decreased" to indicate the effect on the **Miss Rate** for the code above: [4 pt]

Use `float` instead _____

Half the cache size _____

Split the loop body into: _____

```
data[i] /= 10;
data[i] /= 10;
```

No-write allocate _____

(E) Assume it takes 100 ns to get a block of data from main memory. If our L1 data cache has a hit time of 2 ns and a miss rate of 3%, what is the average memory access time (AMAT)? [1 pt]

_____ ns

2: Processes

(A) The following function prints out four numbers. In the following blanks, list three possible outcomes: [3 pt]

```
void concurrent(void) {
   int x = 3, status;
   if (fork()) {
      if (fork() == 0) {
         x += 2;
         printf("%d",x);
      } else {
         wait(&status);
         wait(&status);
         x -= 2;
      }
   }
   printf("%d",x);
   exit(0);
}
```

(1) _____

(2) _____

(3) _____

(B) For the following examples of exception causes, write "**N**" for intentional or "**U**" for unintentional from the perspective of the user process. [2 pt]

System call _____ Hardware failure _____

Segmentation fault _____ Mouse clicked _____

(C) Briefly define a **zombie** process. Name a process that can *reap* a zombie process. [2 pt]

Zombie process:

Reaping process:

(D) In the following blanks, write "**Y**" for yes or "**N**" for no if the following need to be updated when **execv** is run on a process. [2 pt]

Page table _____ PTBR _____ Stack _____ Code _____