CSE 351 Spring 2021 – Unit Summary #3 – Task 3
Due Fri 5/28/21 11:59pm to Gradescope

Your Name:_____

UWNet ID (email):_____

Academic Integrity Statement_____

All work on these questions is my own. I have not shared or discussed my answers with anyone else. (please sign) (1 point)

- To complete Task 3, please either:
  - print these THREE pages, fill them out and then scan and convert into a pdf
  - use digital ink or otherwise annotate the pdf electronically
- Gradescope requires you to upload a pdf
- Fill in your name and UW NetID above, then read the Academic Integrity Statement and sign your name indicating that you understand and will comply with the statement. If you are not printing this out or do not have access to digital ink, just type your full name.
- You may show scratch work for potential partial credit but showing work is not required. Be sure your final answer is placed 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, communicating with anyone other than the course staff, etc.)
- If you have questions, please ask on the Ed Board. A private post is fine! Questions about the unit summaries will not be answered in office hours.
- Refer to the Unit Summary webpage for additional information: https://courses.cs.washington.edu/courses/cse351/21sp/unit_summaries/

Good Luck!

1. Cache parameters (3 points)

You have a byte-addressed machine with 256 KiB of Physical address space. You have a 8-way associative L1 data cache of total size 2048 bytes with a cache block size of 64 bytes.

a) [2 pt] Give the number of **bits** needed for each of these:

Cache Block Offset: _____ Cache Tag: _____

b) [1 pt] How many **sets** will the cache have? _____

2. Structs (5 points)
For this question, assume x86-64 and the following C struct definition.

```
typedef struct {
    char* name;
    short servings;
    char rating;
    char* ingredients[6];
    float cost;
} recipe;
```

a) [1 pt] What is the byte offset where **`rating`** begins?

b) [1 pt] What is the byte offset where **`ingredients[3]`** begins?

c) [1 pt] Is there any **internal fragmentation**? If so, how many bytes and where?

YES / NO If yes, number of bytes _____, where _____

d) [1 pt] Is there any **external fragmentation**? If so, how many bytes and where?

YES / NO If yes, number of bytes _____, where _____

e) [1 pt] Can the compiler reduce the amount of fragmentation? (circle one)?

YES / NO

3. Cache hit rate (12 points)

a) [4 pts] You have a direct mapped cache containing 128 bytes with a cache block size of 32 bytes. The cache uses LRU replacement and write-allocate and write-back policies. Assume **`i`** and **`j`** are stored in registers, and that the array **`happy`** starts at address 0x0. Give the **hit** rate (as a fraction or a %) for the following two loops. Assume the cache starts out empty.

```
#define LEAP 4
#define SIZE 64
int happy[SIZE];
... // Assume happy has been initialized to contain values.
... // Assume the cache starts empty at this point.
for (int i = 0; i < SIZE; i += LEAP) {       // Loop 1
  happy[i] = happy[i] + i * (i + 2);
}
for (int j = 1; j < SIZE; j += (LEAP * 2)) { // Loop 2
  happy[j] = happy[j] + j * 5;
}
```

Hit Rate for Loop 1: _____ **Hit** Rate for Loop 2: _____

b) [8 pts] For each of the changes proposed below, indicate how it would affect the **hit rate** of each loop above in part c) *assuming that all other factors remained the same* as they were in the original problem. Circle one of: "increase", "no change", or "decrease" for each loop.

Change associativity from direct mapped to two-way:
Loop 1: increase / no change / decrease
Loop 2: increase / no change / decrease

Change **`LEAP`** from 4 to 8:
Loop 1: increase / no change / decrease
Loop 2: increase / no change / decrease

Change cache size from 128 bytes to 256 bytes:
Loop 1: increase / no change / decrease
Loop 2: increase / no change / decrease

Change block size from 32 bytes to 16 bytes:
Loop 1: increase / no change / decrease
Loop 2: increase / no change / decrease

4. Processes (5 points)
The following function prints out numbers.

```
void summer(void) {
  int x = 3;
  if (fork()) {
    if (fork()) {
      x += 7;
      fork();
     }
  } else {
    x += 2;
  }
  printf("%d ", x);
  if (fork()) {
    x -= 6;
  } else {
    x -= 1;
    printf("%d ", x);
    fork();
    printf("Bye ");
  }
  exit(0);
}
```

a. [1 pts] What is the **total number of processes created** (including the original process that called **`summer`**) by this function?

b. [1 pt] Is it possible for the numbers that are printed to appear in **descending/non-increasing** order (highest value to lowest value) in the output?

YES / NO

c. [1 pt] How many times will "Bye" be printed?

d. [1 pt] What is the **smallest** number that will be printed?

e. [1 pt] What is the **largest** number that will be printed?