Question F6: Structs [10 pts]

For this question, assume a 64-bit machine and the following C struct definition.

```c
typedef struct {
    char* title; // title (e.g. "HW SW INTERFACE")
    char dept[3]; // dept (e.g. "CSE")
    short num; // course number (e.g. 351)
    int enrolled; // students enrolled
} course;
```

(A) How much memory, in bytes, does an instance of `course` use? How many of those bytes are \textit{internal} fragmentation and \textit{external} fragmentation? [6 pt]

<table>
<thead>
<tr>
<th>sizeof(course)</th>
<th>Internal</th>
<th>External</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(B) Assume that an instance `course c` is allocated on the stack and an array `char ar[]` is allocated 40 bytes below `c` (i.e. `&ar + 0x28 == (char*)&c`). Fill in the blanks below with the new ASCII characters stored in `c.dept` after the following loop is executed. \textbf{Hint}: recall that the values 0x30 to 0x39 correspond to the ASCII characters '0' to '9'. [4 pt]

```c
for (int i = 0; i < 52; ++i) {
    ar[i] = i;
}
```

<table>
<thead>
<tr>
<th>c.dept[0]:</th>
<th>'___'</th>
</tr>
</thead>
<tbody>
<tr>
<td>c.dept[1]:</td>
<td>'___'</td>
</tr>
<tr>
<td>c.dept[2]:</td>
<td>'___'</td>
</tr>
</tbody>
</table>
19sp Final

1. Caches (15 points total)

You are using a byte-addressed machine with 64 KiB of Physical address space. You have a 2-way associative L1 data cache of total size 256 bytes with a cache block size of 16 bytes. It uses LRU replacement and write-allocate and write-back policies.

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? ___________

c) [4 pts] Assume i and j are stored in registers, and that the array x starts at address 0x0. Give the miss rate (as a fraction or a %) for the following two loops, assuming that the cache starts out empty.

```c
#define LEAP 2
#define SIZE 128
int x[SIZE];
... // Assume x has been initialized to contain values.
... // Assume the cache starts empty at this point.
for (int i = 0; i < SIZE; i += LEAP) {
    // Loop 1
    x[i] = x[i] + i * i;
}
for (int j = 1; j < SIZE; j += LEAP) {
    // Loop 2
    x[j] = x[j] + j * 2;
}
```

**Miss Rate for Loop 1:** ___________  **Miss Rate for Loop 2:** ___________

d) [8 pts] For each of the changes proposed below, indicate how it would affect the **miss 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 2-way to direct mapped: | Loop 1: increase / no change / decrease | Loop 2: increase / no change / decrease |
| Change LEAP from 2 to 4: | Loop 1: increase / no change / decrease | Loop 2: increase / no change / decrease |
| Change cache size from 256 bytes to 512 bytes: | Loop 1: increase / no change / decrease | Loop 2: increase / no change / decrease |
| Change block size from 16 bytes to 32 bytes: | Loop 1: increase / no change / decrease | Loop 2: increase / no change / decrease |
Question F8: Processes [18 pts]

(A) The following function prints out four numbers. In the following blanks, list three possible outcomes: [6 pt]

```c
void concurrent(void) {
    int x = 2, status;
    if (fork()) {
        x *= 2;
    } else {
        x -= 1;
    }
    if (fork()) {
        x += 1;
        wait(&status);
    } else {
        x -= 2;
    }
    printf("%d", x);
    exit(0);
}
```

1. ____________  
2. ____________  
3. ____________

(B) For each of the following types of synchronous exceptions, indicate whether they are intentional (I) or unintentional (U) AND whether they are recoverable (R), maybe recoverable (M) or not recoverable (N). [6 pt]

<table>
<thead>
<tr>
<th>Type</th>
<th>I/U</th>
<th>R/M/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trap</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fault</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Abort</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) For the following scenarios, circle the outcome when the child process executes `exit(0)`. [6 pt]

<table>
<thead>
<tr>
<th>Scenario:</th>
<th>Outcome for child:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent is still executing.</td>
<td>Alive</td>
</tr>
<tr>
<td>Parent has called <code>wait()</code></td>
<td>Alive</td>
</tr>
<tr>
<td>Parent has terminated.</td>
<td>Alive</td>
</tr>
</tbody>
</table>
(a) Virtual memory is an extremely powerful abstraction with many benefits. For each benefit listed below, provide a brief (1-2 sentence) explanation of how VM accomplishes it.

i. (4 points) Protecting processes from one another.

ii. (4 points) Allow programs to use more memory than exists on the machine.

(b) Last month, I turned off my desktop computer, opened it up, added some more RAM (doubling it from 16GiB to 32GiB), closed it, and turned it back on.

i. (4 points) Did the size of the virtual address space change? Why or why not?

ii. (4 points) Did the size of the physical address space change? Why or why not?

iii. (4 points) After the RAM upgrade, I was able to have more programs open (and thus using memory) before performance began to degrade. Why did this happen? What was causing performance to drop after a certain number of programs were running? (1-2 sentences)
(c) Imagine the following system:
- 16-bit virtual addresses, 10-bit physical addresses.
- A page size of 16 bytes.
- 2-way set associative TLB with 8 total entries.
- All page table entries NOT in the initial TLB start as invalid.

i. (4 points) Compute the following quantities

- Page offset bits: __________
- PPN bits: __________
- VPN bits: __________
- TLB index bits: __________

ii. (6 points) The TLB has the following state:

<table>
<thead>
<tr>
<th>Set</th>
<th>Tag</th>
<th>PPN</th>
<th>Valid</th>
<th>Tag</th>
<th>PPN</th>
<th>Valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0xB2</td>
<td>–</td>
<td>0</td>
<td>0x283</td>
<td>0x3A</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0xF2B</td>
<td>0x29</td>
<td>1</td>
<td>0x0E8</td>
<td>0x1D</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0x004</td>
<td>–</td>
<td>0</td>
<td>0x346</td>
<td>–</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0x3F4</td>
<td>0x1B</td>
<td>1</td>
<td>0x257</td>
<td>0x36</td>
<td>1</td>
</tr>
</tbody>
</table>

Fill in the associated information for the following accesses to virtual addresses. (enter n/a if the answer cannot be determined):

<table>
<thead>
<tr>
<th>Virtual Address</th>
<th>TLB Hit?</th>
<th>Page Fault?</th>
<th>PPN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x3A17</td>
<td>_____</td>
<td>_____</td>
<td></td>
</tr>
<tr>
<td>0x0123</td>
<td>_____</td>
<td>_____</td>
<td></td>
</tr>
</tbody>
</table>
Question F10: Memory Allocation [18 pts]

(A) In the following code, briefly identify the TWO memory issues and their fixes. [6 pt]

```
int N = 32;
long* func(long src[]) {  
    long* p = (long*) malloc(N * sizeof(long));
    for (int i = 0; i < N; i++) {  
        p[i] += src[i];
    }
}
```

Error 1: Using uninitialized memory in p[i].
Fix 1: Replace malloc with calloc or change p[i] += src[i] to p[i] = src[i];

Error 2: Memory leak — no way to access malloc’ed memory once func returns.
Fix 2: Add return p at end of func.

(B) We are using a dynamic memory allocator on a 64-bit machine with an explicit free list, 16-byte boundary tags, and 8-byte alignment. Assume that a footer is always used. [6 pt]

<table>
<thead>
<tr>
<th>Request</th>
<th>block addr</th>
<th>return value</th>
<th>block size</th>
<th>internal fragmentation in this block</th>
</tr>
</thead>
<tbody>
<tr>
<td>p = malloc(9);</td>
<td>0x628</td>
<td>0x_____</td>
<td>_____ bytes</td>
<td>_____ bytes</td>
</tr>
</tbody>
</table>

(C) Consider the C code shown here. Assume that the malloc call succeeds and that all variables are stored in memory (not registers). In the following groups of expressions, circle the one whose returned value (assume just before return 0) is largest. [6 pt]

```
#include <stdlib.h>
long glob = 10;
char* str = "351";

int main() {  
    short* sp = malloc(8);
    int ONE = 1;
    free(sp);
    return 0;
}
```

Group 1: &sp  sp  &str
Group 2: &glob main  str
Group 3: glob  ONE  *str

End of Exam
Did you write your Student ID Number on the top-right corner of every odd page?