CSCI 356 Fall 2017 : Practice Exam II (The Practicing)

DO NOT OPEN EXAM PACKET UNTIL INSTRUCTED TO DO SO

YOU MAY FILL IN INFORMATION ON THE FRONT NOW

PLEASE TURN OFF ALL ELECTRONIC DEVICES

ID#:	
Name:	

- This exam is closed book. You are allowed one (1) 8.5" x 11" handwritten note sheet
- You will have eighty (80) minutes to complete this exam.
- Answer the questions only in the spaces provided on the question sheets.
- If you give multiple solutions to a problem without indicating which one you want graded, the grader may select one to grade.
- Your answers do not need to be complete, grammatically correct sentences.

Problem	Points	Possible
1		
2		
3		
4		
5		
6		
7		
Total		

1. Suppose we have a cache memory with the following properties:
   • The cache size (C) is 512 bytes (contains 512 data bytes)
   • The cache uses an LRU (least-recently used) policy for eviction. Note that this is the same policy you implemented for project five.
   • The cache is initially empty.

Suppose that for the following sequence of addresses sent to the cache, 0, 2, 4, 8, 16, 32, the hit rate is ⅓. What is the block size (B) of the cache?
(It is a power of two)

It ends up at 8. If you try 4, you get a rate of ⅙ (M H M M M M) and if you try 16 you'll get ½ (M H H H M M). At 8, you get ⅓ (M H H M M M)

2. Consider an SRAM-based cache for a DRAM-based main memory. Recall that DRAM is typically 10x slower than SRAM. Neglect the possibility of other caches or levels of the memory hierarchy below main memory. If a cache is improved, increasing the typical hit rate from 98% to 99%, by what percent would the typical access time likely decrease?

8 + ⅓ % improvement

3. What section of memory holds the assembly code for `printf`?
   (a) Stack
   (b) Heap
   (c) Shared Libraries
   (d) Kernel Memory

4. For each of the following code snippets, write down all symbols in the resulting object files from compilation. Write whether it is a weak global, strong global, or local variable, and what section of the final compiled ELF binary the variable will go into. Fill in the value if you have enough information to determine the value.

main.c	**foo.c**

`#include<stdio.h>`
 	
`int x;`
`int y;`
`int z = 0;`
`int main()`
`{`
`  printf("%x\n", x);`
`  printf("%x\n", y);`
`  x = 0xdecafbad;`
`  printf("%x\n", x);`
`  printf("%x\n", y);`
`  return 0;`
`}`
	`short x = 5;`
	`short y = 2;`

1.

File	Symbol	Strength and Scope	Value	ELF Section
main.o	x	weak global	-	.data
	y	weak global	-	.data
	z	strong global	0	.data
	main	strong global	-	.text
foo.o	x	strong global	5	.data
	y	strong global	2	.data
	(unused: foo.o has only two symbols)			

5. We are given the following code. The left hand side is the original code, while the code on the right is a re-write of the same code.

```
2: int calcHash(char *str) {
3:    unsigned int i;
4:    int hash = 0;
5:
6:    for(i = 0; i < strlen(str); i++) {
7:       hash += str[i] * 32 + i;
8:
9:    }
10:
11:
12:
13:
14:
15:    return hash;
16: }
```

```
2: int calcHash(char *str) {
3:    unsigned int i, len = strlen(str);
4:    int hashA = 0, hashB = 0;
5:
6:    for (i = 0; i < len - 1; i += 2) {
7:       hashA += (str[i] << 5) + i;
8:       hashB += (str[i + 1] << 5) + i + 1;
9:    }
10:
11:   if (i == len - 1) {
12:      hashA += (str[i] << 5) + i;
13:   }
14:
15:   return hashA + hashB;
16: }
```

a. Explain in one or two sentences how moving `strlen(str)` from line 6 to line 3 improves the performance of this code.

Avoids repeated computation

b. Would this transformation would preserve the exact functionality of the original code? Explain.

I would accept two answers here:

i. Yes, because it will return the same answer each time and not change any program state otherwise.

ii. No, because we don't know if strlen() has any side effects [it turns out it doesn't, but you might not be expected to know that offhand]

c. Could a compiler perform this optimization? Why or why not?

No; see ii above.

d. Point out one other optimization that was added to this code and explain how it improves performance.

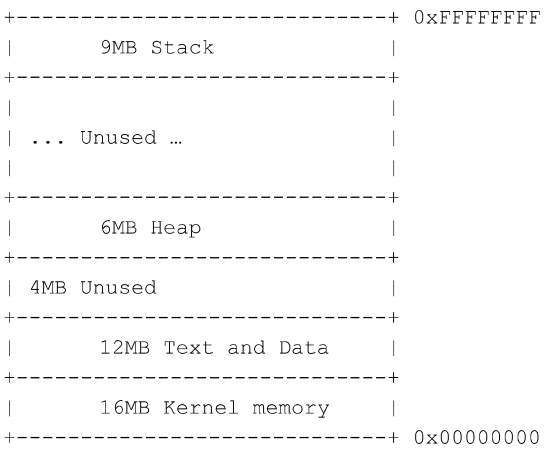
Loop unrolling. Improves pipeline and avoids the prediction necessities.

6. Explain how file descriptors are used for redirection in the following command:

./hex2raw < input.txt > output.txt

(students worried on this one should read up on how dup2 works)

7. Consider a process with the following sections of memory in its address space:



Fill in each entry of the page directory on the next page with the name of the corresponding section of memory. The sections are: unallocated, stack, heap, text and data, or kernel memory.

Index	Entry
0 1 2 3	Kernel Memory
4 5 6	Text and Data
7	Unallocated
8 9	Heap
10 11 12 … n-12 n-11 n-10 n-9 n-8 n-7 n-6 n-5 n-4	Unallocated
n-3 n-2 n-1	Stack