CS240 Midterm Solution 07/10/2017

P1(a) 12 pts

Preprocessor.
Compiler.
Linker.
6 pts

Source files that are not modified need not be recompiled when coding/
updating
large C programs.
3 pts

make keeps track of when files have been last modified using operating system
time stamps. If the files on a target's dependency list is up-to-date at the
time
of running make, the target is not regenerated.
3 pts

P1(b) 12 pts

t is assigned the address of s.
3 pts

u is assigned the address of t.
3 pts

**u is the content of the content of u, hence s.
3 pts

printf() prints the content of s, hence the value 7.
3 pts

P1(c) 12 pts

The first printf() outputs 5.5 to stdout since c is a pointer to a
and *c = 5.5 modifies the content of a.
2 pts

The second printf() outputs 5.5 as well.
2 pts

The third printf() outputs the content of b which is an address
(a pointer to float).
3 pts

*b = 4.4 is likely to lead to segmentation fault since the address
contained in b has not been specified (hence may fall outside the
range of addresses assigned to the process running the code).
3 pts

The fourth printf() is not executed since *b = 4.4 leads to segmentation
fault which terminates the process.
2 pts

P2(a) 15 pts

Set up a mask that is 1 in the first bit position (least significant bit) and
0 elsewhere.

2 pts

Shift the input x by 4 bits to the right.
2 pts

AND the shifted result with the mask and output its value.
2 pts

```
main()
{
char x;
unsigned int y;
unsigned char m;

  // read input
  scanf("%c", &x);

  // set mask to 00 ... 01
  m = ~(~0 << 1);

  // shift x by 4 bits
  x = x >> 4;

  // AND x and m
  y = x & m;
  printf("%u\n", y);

}
```

[There is some degree of freedom for choosing variable types that
lead to correct results.]
9 pts

P2(b) 15 pts

float *f(int) means that a f is a function with one integer argument that
returns a pointer to float.
3 pts

float (*g)(int) means that a g is a function pointer with one integer
argument that returns a float.
3 pts

```
g = &f;
(*g)(5);
```

[C is lax in the syntax of function pointers, hence other expressions such
as g = f will work as well.]
4 pts

```
float h(int (*r)(int), int a)
{
  return (*r)(a);
}
```

[There is degree of freedom in coding with function pointers due to syntax
laxity. For example, "return r(a)" will work as well.]
5 pts

P3(a) 17 pts

By accessing a[5], a[6], a[7] (etc.) we accessed memory above the stack frame
of the called function (i.e., callee) which belongs to the caller. That is,

we smashed/corrupted the caller's stack frame.

5 pts

If the return address in the caller's stack frame is overwritten with a value (i.e., address) that does not belong to the running program (i.e., process), then a segmentation fault will occur when the callee tries to return to the caller.

5 pts

If overwritten return address does not trigger segmentation fault but contains code other than the calling function, then the callee will not return to the caller but run other code with negative consequences.

4 pts

gcc adds stack smashing protection code using a bit pattern (i.e., canary) that is changed when overflow (e.g., array int a[5]) occurs. If the return address is surgically modified without corrupting the canary, gcc cannot detect smashing.

3 pts

P3(b) 15 pts

| address | content |
|---|---|
| ... | ... |
| 4008 | 9 |
| 4004 | 8 |
| 4000 | 7 |
| ... | ... |
| 3008 | 6 |
| 3004 | 5 |
| 3000 | 4 |
| ... | ... |
| 2008 | 3 |
| 2004 | 2 |
| 2000 | 1 |
| ... | ... |
| 1508 | 4000 |
| 1504 | 3000 |
| 1500 | 2000 |
| ... | ... |
| 1000 | 1500 |
| ... | ... |

7 pts

*(*d+2) means: since d is address 1000 in the above example, *d+2 is 1500+8 = 1508 since +2 is +8 for type int. Applying another indirection * yields content at address 4000 which is 7.

3 pts

*(*(d+1)+2) means: d+1 = 1504, *(d+1) = 3000, +2 yields 3008, and final indirection * gives content 6.

3 pts

```
void read_array(int **);
```
2 pts

```
read_array(d);
```
2 pts

Bonus 10 pts

Executing return will return to the function that called the function where
a file is being opened. exit(1) will terminate the process whose file open
failed.
5 pts

Oftentimes when opening a file fails, it makes no sense for a program to
continue
and it should terminate. If a return statement is used, a process does not
terminate but return to the caller of the current function. By calling
exit(1),
a process is terminated when failure to open a file is detected and an
exit condition (value 1) is conveyed to the parent process to indicate that
the process terminated in an irregular manner.
5 pts