CAS CS 210 - Computer Systems
Fall 2014

Problem Set 1 (PS1) ('C' Basics, Logic and Data Representaiton)
Out: September 09
Due: Part A September 18, 1:30 pm; Part B September 23, 1:30 pm

NO LATE SUBMISSIONS WILL BE ACCEPTED

To be completed individually. For all questions, show your work in obtaining the final answers.

PART A: 'C' Basics and Logic

READ: K&R: chapter 1, 5.1-5.6, 6.1-6.4

1) hello.c

```
1 #include<stdio.h>
2
3 int
4 main(int argc, char **argv)
5 {
6   printf("Hello World\n");
7   return 0;
8 }
```

Describe the meaning/side effect of each non-blank line.

2) Memory and Pointers

```
1  #include <stdio.h>
2
3  int myint;
4  int *ip;
5  char mystring[6] = "hello";
6  char *cp;
7
8  int myfunc0(int x, int *ip)
9  {
10   *ip = *ip + 2;
11   x++;
12   return x;
13 }
14
15 int myfunc1(char *c)
16 {
17   if (*c >= 'a' && *c <= 'z') {
18     *c +='A' - 'a';
19     return 1;
20   }
21   return 0;
22 }
23
24 int main(int argc, char **argv)
25 {
26   myint = 0;
27   ip = &myint;
28   cp = mystring;
29
30   while (myfunc1(cp)) {
31     printf("%d\n", myfunc0(myint, ip));
32     cp++;
33   }
34
35   printf("mystring:%s  myint:%d\n", mystring, myint);
36   return 0;
37 }
```

Please provide the output below for the program listing.

Please fill in the missing values for the following table assuming that we stop the program just prior to it exiting at line 36. All address values should be written as 8 digit hex values and all integer values as simple decimals.

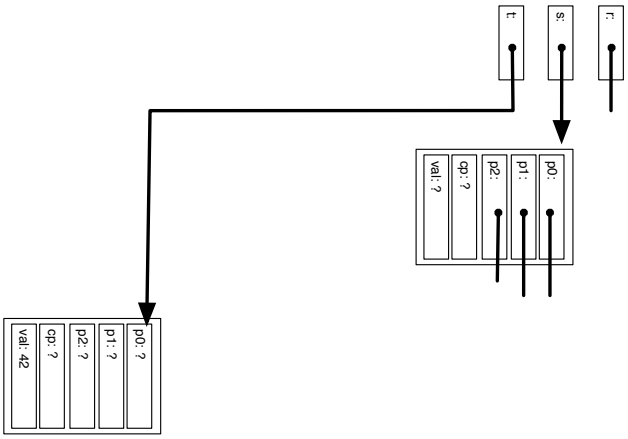
Name	Address	Value
mystring[0]	0x0804972c	
mystring[1]	0x0804972d	
mystring[2]	0x0804972e	
mystring[3]	0x0804972f	
mystring[4]	0x08049730	
mystring[5]	0x08049731	
ip	0x0804973c	
myint	0x08049740	
cp	0x08049744	

3) Pointers and Structs

```
1  struct myStruct {
2    struct myStruct *p0;
3    struct myStruct *p1;
4    struct myStruct *p2;
5    char            *cp;
6    int              val;
7  };
8
9  struct myStruct *r;
10 struct myStruct *s;
11 struct myStruct *t;
12
13 s = malloc(sizeof(struct myStruct));
14 s->p0 = malloc(sizeof(struct myStruct));
15 s->p0->p2 = malloc(sizeof(struct myStruct));
16 s->p0->p2->p1 = malloc(sizeof(struct myStruct));
17 s->p0->p2->p0 = s->p0;
18 r=s->p0->p2->p1;
19 s->p1 = malloc(sizeof(struct myStruct));
20 r->p0 = malloc(sizeof(struct myStruct));
21 r->p0->p2 = malloc(sizeof(struct myStruct));
22 r->p0->p2->p2 = malloc(sizeof(struct myStruct));
23 t = malloc(size(struct myStruct));
24 s->p0->p2->p1->p0->p2->val = 21;
25 t->val = 42;
26 s->p2 = t;
27 s->p0->p2->p0->p2->p1->p0->p2->val = 3;
```

A struct is a multi-byte programmer defined type that groups together several members (K&R ch 6). Sizeof can be used to determine the aggregate number of bytes that a particular struct type requires. Malloc is a standard 'C' library call that dynamically allocates the requested number of bytes of memory (K&R 7.8.5). Malloc returns the address of the newly allocated memory. Storing the address in a variable of the appropriate pointer type allows you to access the memory allocated by malloc. In the case of a struct pointer you use the '->', called the member selection operator, to access a particular field of the struct pointed too. For futher details on structs and malloc see the appropriate sections in K&R.

Complete the diagram on the next page. Illustrate the side effect of the above code fragment. Draw all additional boxe and complete and add arrows as needed. Note assume there are no failures in calls to malloc. You may use '?' to indicate unknown values of fields. Be sure to indicate all instances of the struct and all field values.


t:
s:
r:
val: ?
cp: ?
p2:
p1:
p0:
val: 42
cp: ?
p2: ?
p1: ?
p0: ?

4) Logic Gates and Truth Tables)

1. Prove both of DeMorgan's Laws using Truth Tables. DeMorgan's Laws are:

   (a) Law 1: Stated in english and in 'C'

   English: Not A and B is the same as Not A or Not B
   'C': $!(A \&\& B) == (!A || !B)$

   (b) Law 2: Stated in english and in 'C'

   English: Not A or B is the same as Not A and Not B
   'C': $!(A || B) == (!A \&\& !B)$

2. Only using NOT, OR and AND gates draw the logic gate cicuit for the following boolean expression. $(w||!x)\&\&(y||(x\&\&z))$

PART B: Data Representation

1) Book problems

1. Solve problem 2.61, on page 121, from our CS:APP2e text.
2. Solve problem 2.68, on page 123, from our CS:APP2e text.
3. Solve problem 2.71, on page 124, from our CS:APP2e text.
4. Solve problem 2.76, on page 126, from our CS:APP2e text.

2) 2's Complement Respresentation

Fill in the below table assuming a 32 bit computer that uses 2's complement representation, `INT_MAX` and `INT_MIN` are defined as the computer's signed integer representation maximum and minimum value respectively, and:

```
int x = -1, y = 0xfeedface, z = INT_MAX, i = sizeof(short *);
```

C Expression	Hexadecimal
`x`	
`y`	
`z`	
`i`	
`z<<3`	
`z<<((i>>1)-1)`	
`~0 == (z + INT_MIN)`	
`y & 0xffff`	
`y >> 16`	
`(y >> 16) | 0xffff`	
`(~(0x10>>2)+1) == (x*i)`	
`(~z+1) + -1`	
`(~((~x) << 1)) & y`	
`((y<<3)+INT_MIN)^((y<<3)+INT_MIN)`	

3) Misc

1. $(010101)_2$ to base 10
2. $(011100)_2$ to base 16
3. $(54.125)_{10}$ to binary
4. $(122.3)_8$ to base 16
5. Find the decimal equivalent of the five-bit twos complement number: 11111
6. Show the results of adding the following pairs of six-bit twos complement numbers in decimal and indicate whether or not overflow occurs for each case.
   (a) 111110 + 111101
   (b) 110111 + 110111
   (c) 111111 + 001011
7. Complete the following table for the 5-bit 2's complement representation. Show your answers as signed base 10 decimal integers and the 2's complement binary value.

value	decimal	binary
Largest Positive Number		
Most Negative Number		
Number of distinct Numbers		