ECE 131 Programming Fundamentals – Exam #3a

Fall, 2011 **Closed book, closed notes**

Last Name:

First Name:

10 pts. 1. Consider the following C declaration and variable declaration/initialization:

```
struct date {
   int year;
   int month;
   int day;
};
struct time dd = {2011, 12, 1};
```

Which one of the following statements sets the value of dd to represent the date December 25, 2011?

(a) `dd->day = 25;`
[✓ (b)] `dd.day = 25;`
(c) `dd.day = {2011, 12, 25};`
(d) `dd = dd + {0, 0, 24};`

10 pts. 2. Consider the following C declaration and variable declarations/initializations:

```
struct date {
   int year;
   int month;
   int day;
};
struct date dd = {2011, 12, 1}, *pd = &dd;
```

Which one of the following statements sets the value of dd to represent the date December 25, 2011?

[✓ (a)] `pd->day = 25;`
(b) `pd.day = 25;`
(c) `pd.day = {2011, 12, 25};`
(d) `pt->day = dd.25;`

10 pts.

3. The following program should print "odg" but prints "hotdogs" instead. Make the simple correction in the copyOdds() function that will cause it to work correctly. Hint: this is one of the most common errors in C programming.

```
int main()
{
   void copyOdds(char source[], char result[]);
   char buffer[80];
   copyOdds("hotdogs", buffer);
   printf("%s\n", buffer);
   return 0;
}

void copyOdds(char source[], char result[])
{
   char ch;
   int i = 0, j = 0, odd;
   while (source[i]) {
      odd = i % 2;
      if (odd = 1) {          ==
         result[j] = source[i];
         j++;
      }
      i++;
   }
   result[j] = '\0';
}
```

10 pts. 6. Given the following variable declaration in C:

```
float data[4];
```

Circle the one item in the following list that is not the same type as "pointer-to-float":

```
data

&data[3]
```

[✓ `data[2]`]

```
&data[1] - 1
```

10 pts. 7. The following program (based on Exercise 11.2) should print 100, 123, 200 but instead prints 100 and stops. Correct the insertEntry() function to make it work correctly. Hint: you should still end up with only two lines of code in insertEntry().

```
#include <stdio.h>

struct entry
{
   int value;
   struct entry *next;
} ;

void printList(struct entry *list)
{
   while (list != NULL)
   {
      printf("%d ", list->value);
      list = list->next;
   }
   printf("\n");
}

void insertEntry(struct entry *newEntry, struct entry *afterWhich)
{
   ~~afterWhich->next = newEntry->next;~~
   newEntry->next = afterWhich->next;
}  afterWhich -> next = newEntry;

int main(void)
{
   struct entry newEntry = {123, NULL};
   struct entry b = {200, NULL};
   struct entry a = {100, &b};

   insertEntry(&newEntry, &a);

   printList(&a);

   return 0;
}
```

10 pts. **8. Given the following variable definitions in C:**

```
char bigfan[] = "Lobo Louie loves Lobo Lucy!";
char *bf = bigfan;
```

What character is referenced by each of the following? (Five answers are called for)

`*bigfan`	L
`*bf`	L
`bigfan[5]`	L
`*(bf + 7)`	u
`bf[13]`	v

10 pts. **9. Given the following variable definitions in C:**

```
unsigned char x = 0x3A;        00111010
unsigned char y = 0x5F;        01011111
unsigned char z;
```

What does z evaluate to in each of the following?

`z = x & y;`	00011010 = 0x1A = 26
`z = x | y;`	01111111 = ~~0x7A = 122~~ 0x7F = 127
`z = x ^ y;`	01100101 = 0x65 = 101
`z = ~x;`	11000101 = 0xC5 = 197

10 pts. **10. What does the following program print?**

```
main() {
   unsigned char x = 0x54;
   printf("x >> 2 = 0x%02x\n", x >> 2);
   printf("x << 3 = 0x%02x\n", x << 3);
   printf("swap(x) = 0x%02x\n", (x << 4) | (x >> 4));
}
```

0x54 = 01010100
>>2 = 00010101 = 0x15
<<3 = 10100000 = 0xA0

x >> 2 = 0x15;
x << 3 = 0xA0;
swpp(x) = 0x45;

10 pts. **4. What does the following program print?**

```
int main()
{
   void something(char source[], char result[]);
   char buffer[80];
   something("hotdogs", buffer);
   printf("%s\n", buffer);
   return 0;
}

void something(char source[], char result[])
{
   int i = 0, n = 0;
   while (source[i++] != '\0')
      n++;
   for (i = 0; i < n; i++)
      result[n - 1 - i] = source[i];
   result[n] = '\0';
}
```

"something" reverses the string.

get string length to n

i	n	source[i]	result[n-1-i]
0	1	h	
1	2	o	
2	3	t	
3	4	d	
4	5	o	
5	6	g_s	
6	7		
7		NUL	

i	n-1-i	soure[i]	result[n-1-i]
0	6	h	resut[6]=h
1	5	o	[5]=o
2	4	t	[4]=t
3			
i	Δ	s	

[0]=s
[7]=nul

$\boxed{\text{sgodtoh}}$

10 pts. **5. A palindrome is a string that, when reversed, is the same string. For instance, "noon" is a palindrome but "noone" is not. The following function should return 1 when given a palindrome and 0 if the string is NOT a palindrome. In the if statement, replace the XXXX with an expression that will cause the function to perform correctly.**

Suggestion: if this one isn't obvious to you, come back to it after you've done the other problems.

```
int isPalindrome(char source[])
{
   int i = 0, n = 0;
   while (source[i++] != '\0')
      n++;
   for (i = 0; i < n/2; i++)
      if (source[i] != source[XXXX])
         return 0;
   return 1;
}
```

Problem 4 gives a big help here. I showed how to traverse a string backward.

$\boxed{XXXX = n-1-i}$