Fall 2010 Final Exam

I. A. (i). flurg.i contains the c-code stripped of comments and headers. The process is called pre-processing.

(ii). flurg.s contains the code in assembly language. This process is called compilation.

(iii). flurg.o contains the code in machine language. This is called assembly.

(iv). flurg contains the code in machine language. but in the proper order so it runs successfully. This process is called linkage.

(v). flurg.s

B. Access time is shorter and there is less memory associated with each type

C+D. Did we ever learn what they are?

E. The L2 cache

II. A. (i). 0x6B = 0b0110 1011

~i = 0b1001 0100

$\boxed{\sim i = 0x94}$

(ii). 0x6B = 0b0110 1011

!i = 0x00

$\boxed{!!i = 0x01}$

(iii). 0x6B = 0b0110 1011   0xA7 = 1010 0111

i|j = 0b1110 1111

$\boxed{i|j = 0xEF}$

(iv). 0x6B = 0b0110 1011   0xA7 = 0b1010 0111

j & 0x0F = 0b1010 0111 & 0b0000 1111 = 0b0000 0111

i < (j & 0x0F) = i < 0b0000 0111 = F = $\boxed{0x00}$

(v). 0x6B = 0b0110 1011

i<<2 = 0b1010 1100

$\boxed{i<<2 = 0xAC}$

B. 0x0000

C. 0x7FFF

D. 0xFFFF

E. 0x8000

```
F.   1 1 0 1 1 0 1 1 1
   +   1 1 1 0 0 1 0 1
   -------------------
   1 0 0 0 1 1 1 0 0
```

```
G.   A 5 9 C 2
   +   2 B 8 0
   -----------
     A 8 5 3 F
```

H. $6.25_{10} = 0\ 000000\cancel{00}\ 000000000000000\cancel{110}\ 01_2$

$6 = 0110$

$0.25 = 01$

III. A. p is a void pointer, but points to an integer

B. p is a pointer that points to an array of ints

C. p is a pointer that points to an array of 5 ints

IV. A. A[0] = 10

B. s.x = 70

C. *(s.p) = 666

D. s.s = lou exam?

pointers

V. If I have the following:

```
int main(void)
{
  int a=10;
  int b=20;
  int *p=&a;
  char *cp=(char*)&a;

  ...
  (*p)++;
  cp+=4;
  p+=2;
```

and memory is laid out like this:

a	1000	11
b	1004	20
p	1008	1008
cp	1012	1004

what do you see if you print:

(1 point) (A) p

(1 point) (B) *p

(1 point) (C) &p

(1 point) (D) cp

(1 point) (E) (int)(*cp)

(1 point) (F) &cp

(A) 1008

(B) 1009

(C) 1008

(D) 1004

(E) 11

(F) 1012

points: ______
out of a possible 8

exam continues...

VI. What is the value of each of the following assuming that the array A begins at address 1000 and ints are 4 bytes?

```
typedef struct {
   int x;
   int y;
   char s[12];
} Stuff;

int A[500];

int *ip=&A[0];
char *cp=&A[0];
Stuff *sp=&A[0];

ip+=10;
cp+=3;
sp+=5;
```

(1 point) (A) ip

(A) 40

(1 point) (B) cp

(B) 12

(1 point) (C) sp

(C) 20

(5 points) VII. For each of the following, suppose that %eax contains the value x, %ecx contains y. What's stored in %edx after the each operation?

expression	result
leal 0xC(%eax), %edx	x + 12
leal (%eax,%ecx), %edx	x + y
leal (%eax,%ecx, 4), %edx	(4 · y) + x
leal 5(%eax,%eax,8), %edx	8·x + x + 5
leal 0xA(,%ecx,8), %edx	0 + 8·y + 10

(10 points) VIII. Write a C function equivalent to the following assembly (no credit for an answer containing inline assembly).

```
        .text
        .globl mystery
        .type mystery, @function
mystery:
        ~~pushl %ebp~~
        ~~movl %esp, %ebp~~
        movl 8(%ebp), %eax
        addl $10, %eax
        addl 12(%ebp), %eax
        subl 16(%ebp), %eax
        ~~movl %ebp, %esp~~
        ~~popl %ebp~~
        ret
```

```
int mystery(int x) {
int x = 10;
return x;
}
```

CIS 2107
Computer Systems and Low-Level Programming
Fall 2010
Final Exam

December 16, 2010

Name: ______________________________

Page	Points	Score
1	10	
2	15	
2	11	
4	8	
5	8	
6	23	
8	15	
10	10	
Total:	100	

Instructions
The exam is closed book, closed notes. You may *not* use a calculator, cell phone, etc.

For each of the questions, unless otherwise specified, you can assume the following sizes for C data types:

type	bytes
char	1
short	2
int	4
long	8
float	4
double	8
void*	4

CIS 2107 Final
Computer Systems and Low-Level Programming December 16, 2010

I. Short Answer

(A) **The compilation process**. At times during the semester, we've gone through each of the steps of the compilation process separately.

We did:

- `gcc -E flurg.c` to get `flurg.i`.
- `gcc -S flurg.i` to get `flurg.s`.
- `gcc -c flurg.s` to get `flurg.o`.
- `gcc -o flurg flurg.o` to get the executable file `flurg`.

(1 point) i. Describe what's in `flurg.i`. What do we call the process of translating `flurg.c` into `flurg.i`?

____

____

(1 point) ii. Describe what's in `flurg.s`. What do we call the process of translating `flurg.i` into `flurg.s`?

____

____

(1 point) iii. Describe what's in `flurg.o`. What do we call the process of translating `flurg.s` into `flurg.o`?

____

____

(1 point) iv. Describe what's in `flurg`. What do we call the process of translating `flurg.o` and any necessary libraries into `flurg`?

____

____

(1 point) v. Which component (type of program) decides in which register each C variable will be stored?

v. ______________

(1 point) (B) We've described the storage hierarchy in modern computers as a type of pyramid. Name two things that are generally true the higher up the pyramid we go.

Storage gets smaller, I/O gets faster

____

(2 points) (C) Other than the problem of differing libraries, explain why it is that I can't take a Windows executable, and run it on an Intel Mac or a machine running Linux on Intel. Please do not write, "because they're different operating systems". Be more specific about what the major problems are.

____

____

(1 point) (D) Where do we store an integer return value from a function in x86 assembly?

____

(1 point) (E) Memory allocated with `malloc( )` is stored in what memory segment?

____

II. Data Representation

(A) **Some bit operations** If we have `char i = 0x6B, j = 0xA7;`, what is the result of the following operations? Your answer must be in the form of exactly two hex digits[1].

(2 points) i. ~i

i. ______________

(2 points) ii. !!i

ii. 0x6B

(2 points) iii. i|j

iii. ______________

(2 points) iv. i<(j & 0x0F)

iv. ______________

(2 points) v. i<<2

v. ______________

(1 point) (B) In hex, what is the smallest integer that can be represented by a 16-bit two's complement int?

0x1000 0000 0000 0000

(B) 0x8000

(1 point) (C) In hex, what is the largest integer that can be represented by a 16-bit two's complement int?

0x0111 1111 1111 1111

(C) 0x7FFF

(1 point) (D) In hex, what is the largest integer that can be represented by a 16-bit unsigned int?

(D) 0xFFFF

(1 point) (E) In hex, what is −1 as an 16-bit two's complement int?

(E) 0xFFFF

(1 point) (F) What is $110110111_2 + 11100101_2$ in base 2?

$$\begin{array}{rccccccccc} & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1_2 \\ + & & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1_2 \\ \hline & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}$$

(1 point) (G) What is $A59C2_{16} + 2B8D_{16}$ in base 16?

$$\begin{array}{rccccc} & A & 5 & 9 & C & 2_{16} \\ + & & 2 & B & 8 & D_{16} \\ \hline \end{array}$$

[1] Forget about the possibility of the values being promoted to 32-bits. Just behave as though we're living in the land of 8-bit arithmetic.

(4 points) (H) Do one of the two following floating-point questions. (If you do both, we're going to grade the first one and ignore the second.)

i. How would the number 6.25_{10} be stored in a 32-bit C float variable?

ii. Suppose that we have 0xC0400000 stored in a 32-bit C `float` variable.

1) Is the number positive or negative?

1) ____________

2) What is the bias?

2) ____________

3) What are the bits of the exponent part?

____

4) What are the bits of the mantissa part?

____

5) What floating-point number does this represent?

III. Some tricky declarations

Write a very brief description in English of what is declared. For example, if the question is `int func(int A[])`, you'd write, "func is a function which is passed an array of int and returns an int".

(2 points) (A) `void (*p)(int);`

____

(2 points) (B) `int (*p[ ])( )`

____

(2 points) (C) `int *(*p[5])( )`

____

points: ______
out of a possible 11

exam continues...

IV. Print Me

What is the value of each of the following after `func( )` is called?

```c
#include <string.h>
#include <stdio.h>
#include <stdlib.h>

typedef struct {
  int x;
  int *p;
  char *s;
} Stuff;

void func(int[], Stuff);

int main(void)
{
  int A[5]={10,20,30,40,50};
  int x=60;
  Stuff s;

  s.x=70;
  s.p=&x;
  s.s=malloc(20);
  strcpy(s.s, "long exam?");

  func(A, s);

  return 0;
}

void func(int A[], Stuff s)
{
  s.x=777;
  *(s.p)=666;
  strcpy(s.s, "a little long");
  s.s = malloc(20);
  strcpy(s.s, "not too long");
  A[0]=111;
}
```

(2 points) (A) A[0]

(2 points) (B) s.x

(2 points) (C) *(s.p)

(2 points) (D) s.s

(A) ____________

(B) ____________

(C) ____________

(D) "a little long"

pointers

V. If I have the following:

```
int main(void)
{
  int a=10;
  int b=20;
  int *p=&a;
  char *cp=(char*)&a;

  ...
  (*p)++; increment through the pointer
  cp+=4;
  p+=2;
```

and memory is laid out like this:

a	1000	10	→ 11
b	1004	20	
p	1008	1000	→ 1008
cp	1012	1000	→ 1004

what do you see if you print:

(1 point) (A) p

(1 point) (B) *p

(1 point) (C) &p

(1 point) (D) cp

(1 point) (E) (int)(*cp)

cast what cp points to to a int

(1 point) (F) &cp

(A) 1008

(B) 1008

(C) 1008

(D) 1004

(E) 20

(F) 1012

VI. What is the value of each of the following assuming that the array A begins at address 1000 and ints are 4 bytes?

```
typedef struct {
    int x;     4
    int y;     4     ] 20 bytes
    char s[12]; 12
} Stuff;

int A[500];

int *ip=&A[0];
char *cp=&A[0];
Stuff *sp=&A[0];

ip+=10;
cp+=3;
sp+=5;
```

(1 point) (A) ip

(A) 1040

(1 point) (B) cp

(B) 1003

(1 point) (C) sp

(C) 1100

(5 points) VII. For each of the following, suppose that %eax contains the value x, %ecx contains y. What's stored in %edx after the each operation?

☆ Good God make a cheat sheet for this

expression	result
leal 0xC(%eax), %edx	x + 0xC
leal (%eax,%ecx), %edx	x + y
leal (%eax,%ecx, 4), %edx	x + 4 * y
leal 5(%eax,%eax,8), %edx	(8 * x + x) + 5 = 9 * x + 5
leal 0xA(,%ecx,8), %edx	(8 * y) + 0xA

will be on final (same as problem in slides)

learn what push & pop do? (especially regarding function calls)

difference? { movl (%eax) / movl %eax

(10 points) VIII. Write a C function equivalent to the following assembly (no credit for an answer containing inline assembly).

```
        .text
        .globl mystery
        .type mystery, @function
mystery:
        pushl %ebp
        movl %esp, %ebp
        movl 8(%ebp), %eax
        addl $10, %eax
        addl 12(%ebp), %eax
        subl 16(%ebp), %eax
        movl %ebp, %esp
        popl %ebp
        ret
```

(7 points) IX. Write a function which is passed an int x and returns the number 1 bits which would appear in x's binary representation. Do not assume that an int is necessarily 4 bytes.

```
int  onesInx(int x) {
    unsigned int count = 0;
    while (x>0){
        count += n &1;
        n = n >>1;
    }
    return count;
}
```

(5 points) X. Write a function which is passed a string s. The function removes the last character from s. For example, if s is drinks, after the function finishes, s is drink.

```
void  removeLastChar (char *s ) {
    int len = 0;
    while (s[len] != '\0') {
        len ++;
    }
    s[len -1] = '\0';
}
```

(10 points) XI. Write a function which is passed a pointer to a List, and an int x. The function appends x to the List. You may assume that the pointer to the List is not null. List and list_node are defined as follows:

```
struct list_node {
   int data;
   struct list_node *next;
};
```

```
typedef struct {
  struct list_node *head;
} List;
```

```
void append(List *l, int x){
    struct list_node newNode = {x, NULL};

     struct list_node *current = l->head;
     while (current->next != NULL){
          current = current->next;
      }
      current->next = &newNode;
  }
```

(10 points)XII. Write a function which is passed a sorted int array A[ ], an int len, which is the length of A[ ], and an int x. The function inserts x into A[ ] in its proper sorted position. For example, if A[ ] is {10,30,40,50}, and x is 20, after the function finishes, A[ ] is {10,20,30,40,50}. Do *not* use the [ ] operator.

```
void insert(int A[ ], int len, int x) {
    for (int i=0; i<len; i++){
        if (x> *(A+i)){
            *(A+(i+1))=x;
            return;
        }
    }
```

points: ______
out of a possible ??

question 12 continues. . .