Week 03 Tutorial Questions

1. On a machine with 16-bit `int`s, the C expression `(30000 + 30000)` yields a negative result.

   Why the negative result? How can you make it produce the correct result?

2. Assume that the following hexadecimal values are 16-bit twos-complement. Convert each to the corresponding decimal value.

   i. `0x0013`
   ii. `0x0444`
   iii. `0x1234`
   iv. `0xffff`
   v. `0x8000`

3. Give a representation for each of the following decimal values in 16-bit twos-complement bit-strings. Show the value in binary, octal and hexadecimal.

   i. 1
   ii. 100
   iii. 1000
   iv. 10000
   v. 100000
   vi. -5
   vii. -100

4. What decimal numbers do the following single-precision IEEE 754-encoded bit-strings represent?

   a. `0 00000000 00000000000000000000000`
   b. `1 00000000 00000000000000000000000`
   c. `0 01111111 10000000000000000000000`
   d. `0 01111110 00000000000000000000000`
   e. `0 01111110 11111111111111111111111`
   f. `0 10000000 01100000000000000000000`
   g. `0 10010100 10000000000000000000000`
   h. `0 01101110 10100000101000001010000`

   Each of the above is a single 32-bit bit-string, but partitioned to show the sign, exponent and fraction parts.

5. Convert the following decimal numbers into IEEE 754-encoded bit-strings:

   a. 2.5
   b. 0.375
   c. 27.0
   d. 100.0

6. Write a C function, `six_middle_bits`, which, given a `uint32_t`, extracts and returns the middle six bits.

7. Draw diagrams to show the difference between the following two data structures:

```c
struct {
    int a;
    float b;
} x1;
union {
    int a;
    float b;
} x2;
```

   If `x1` was located at `&x1 == 0x1000` and `x2` was located at `&x2 == 0x2000`, what would be the values of `&x1.a`, `&x1.b`, `&x2.a`, and `&x2.b`?

8. How large (#bytes) is each of the following C `union`s?

   a. `union { int a; int b; } u1;`
   b. `union { unsigned short a; char b; } u2;`
   c. `union { int a; char b[12]; } u3;`
   d. `union { int a; char b[14]; } u4;`
   e. `union { unsigned int a; int b; struct { int x; int y; } c; } u5;`

   You may assume `sizeof(char)` == 1, `sizeof(short)` == 2, `sizeof(int)` == 4.

9. Consider the following C `union`

```
union _all {
   int   ival;
   char cval;
   char  sval[4];
   float fval;
   unsigned int uval;
};
```

If we define a variable `union _all var;` and assign the following value `var.uval = 0x00313233;`, then what will each of the following *printf(3)*s produce:

a. `printf("%x\n", var.uval);`
b. `printf("%d\n", var.ival);`
c. `printf("%c\n", var.cval);`
d. `printf("%s\n", var.sval);`
e. `printf("%f\n", var.fval);`
f. `printf("%e\n", var.fval);`

You can assume that bytes are arranged from right-to-left in increasing address order.

COMP1521 20T2: Computer Systems Fundamentals is brought to you by
the School of Computer Science and Engineering
at the University of New South Wales, Sydney.
For all enquiries, please email the class account at cs1521@cse.unsw.edu.au
CRICOS Provider 00098G