CSE 320 **Fall 2020**

Computer Project #6

Assignment Overview

This assignment develops familiarity with the C programming language, the `gcc` compiler, integer operations, and floating point representation.

It is worth 40 points (4% of course grade) and must be completed no later than 11:59 PM on Thursday, 10/22.

Assignment Deliverables

The deliverables for this assignment are the following files:

`proj06.support.c` – the source code for your library module
`proj06.driver.c` – the source code for your driver module
`proj06.makefile` – the makefile which produces **`proj06`**

Be sure to use the specified file names and to submit them for grading via the CSE handin system before the project deadline.

Assignment Specifications

Some microprocessors do not have floating point hardware. Thus, floating point operations on those systems must be performed using software.

1. You will develop a library module of C functions which operate on floating point values using only integer operations. The library module will include the functions listed below:

```
float get_infinity();       /* Return infinity                 */
float get_nan();            /* Return not-a-number             */
float get_max_normal();     /* Return largest normal           */
float get_min_normal();     /* Return smallest normal          */
float get_max_denormal();   /* Return largest denormal         */
float get_min_denormal();   /* Return smallest denormal        */

int is_negative( float );   /* Test if argument is negative    */
int is_infinity( float );   /* Test if argument is infinity    */
int is_nan( float );        /* Test if argument is not-a-number  */
int is_zero( float );       /* Test if argument is zero        */
int is_denormal( float );   /* Test if argument is denormal    */

float negate( float );      /* Return negation of argument     */
float absolute( float );    /* Return absolute value of argument */
```

The first six functions will return the specified single precision value (all will be positive). The return value from function "get_nan" will have a fraction field were each of the bits is a 1.

The next five functions will return 0 if the specified condition is false, and 1 if the condition is true.

Functions "negate" and "absolute" will perform the appropriate operation on the argument and return the result.

The function declarations (and some auxiliary declarations) are available on the CSE system as:

```
/user/cse320/Projects/project06.support.h
```

Your implementation of the library module will not use any floating point operations, with one exception: any of the functions may use the assignment operation to copy a floating point value from one variable to another.

The functions in your library module will not call any C library functions (such as "printf", "scanf" and "isnan").

2. You will develop a driver module to test your implementation of the library module. The driver module will consist of function "main" and any additional helper functions which you choose to implement.

Your driver module will not be an interactive program, where the user supplies input in response to prompts. Instead, your test cases will be included in the source code as literal constants.

All output will be appropriately labeled.

Assignment Notes

1. The Harris and Harris text contains information about floating point representation and operations (particularly pages 256-259).

2. The source code for the library module and the driver module must be in separate files. And, your source code files must be translated by `gcc`, which is a C compiler and accepts C source statements.

3. You must supply a makefile (named "proj06.makefile"), and that makefile must produce an executable program named "proj06".

4. In order to work with a 32-bit item as both a floating point value and an integer value, you might consider using a union. For example:

```
#include <stdio.h>

union ieee_single
{
  float frep;
  unsigned int irep;
};

int main()
{
  union ieee_single num;
  num.frep = 1.5;

  printf( "Real: %f  Hex integer: %08x \n", num.frep, num.irep );
}
```

When compiled and executed, that program produces the following:

```
Real: 1.500000  Hex integer: 3fc00000
```

The memory location **`num`** can be referenced as an object of type **`float`** (using **`num.frep`**) and an object of type **`unsigned int`** (using **`num.irep`**).