Problem Set 2

Problem 2.1

Determine the size,minimum and maximum value following data types. Please specify if your machine is 32 bit or 64 bits in the answer.

- char
- unsigned char
- short
- int
- unsigned int
- unsigned long
- float

Hint: Use `sizeof()` operator, `limits.h` and `float.h` header files.

Answers:

My machine is 64 bits.

The code is:

```
#include <stdio.h>
#include <limits.h>
#include <float.h>

int main(void)
{
    printf("Print out the size for different data types.\n");
    printf("The size for char is %ld\n", sizeof(char));
    printf("The size for unsigned char is %ld\n", sizeof(unsigned char));
    printf("The size for short is %ld\n", sizeof(short));
    printf("The size for int is %ld\n", sizeof(int));
    printf("The size for unsigned int is %ld\n", sizeof(unsigned int));
    printf("The size for unsigned long is %ld\n", sizeof(unsigned long));
    printf("The size for float is %ld\n", sizeof(float));

    printf("Print out the min and max value for different data types.\n");
    printf("The min value for CHAR = %d\n", SCHAR_MIN);
    printf("The max value for CHAR = %d\n", SCHAR_MAX);
    printf("The min value for UNSIGNED CHAR = %d\n", UCHAR_MAX);
    printf("The min value for SHORT = %d\n", SHRT_MIN);
    printf("The max value for SHORT = %d\n", SHRT_MAX);
    printf("The min value for INT = %d\n", INT_MIN);
    printf("The max value for INT = %d\n", INT_MAX);
    printf("The max value for UNSIGNED INT = %u\n", UINT_MAX);
    printf("The max value for UNSIGNED LONG = %lu\n", ULONG_MAX);
    printf("The min value for FLOAT = %.10e\n", FLT_MIN);
    printf("The max value for FLOAT = %.10e\n", FLT_MAX);
    return 0;
}
```

The output:

ps2

Problem 2.2

Write logical expressions that tests whether a given character variable c is

- lower case letter
- upper case letter
- digit
- white space(includes space, tab, new line)

Answers:

```
#include <stdio.h>
int main()
{
    char c;
    printf("Input a character: \n");
    if (c >= 'a' && c <= 'z')
        printf("%c is a lower case letter.\n", c);
    else if (c >= 'A' && c <= 'Z')
        printf("%c is an upper case letter.\n", c);
    else if (c >= '0' && c <= '9')
        printf("%c is a digit.\n", c);
    else if (c == ' ' || c == '\t' || c == '\n')
        printf("%c is a white space.\n", c);
    else
        printf("No match.\n");
    return 0;
}
```

Problem 2.3

Consider `int val=0xCAFE;` Write expressions using bitwise operations that do the following:

- test if at least three of last four bits(LSB) are on
- reverse the byte order(i.e., produce `val=0xFECA`)
- rotate fourbits(i.e., produce `val=0xECAF`)

Answers:

Ans1:

```c
int result = 0;
result = val & 0xF;
if (result == 0x7 || result == 0xB || result >= 0xD)
    printf("At least three of last four bits(LSB) are on.\n")
```

Ans2:

```c
int result = 0;
result = (val & 0xFF) << 8 | (val >> 8);
```

Ans3:

```c
int result = 0;
result = ((val & 0xF) << 12) & 0xFFFF | (val >> 4);
```

Problem 2.4

Using precedence rules, evaluate the following expressions and determine the value of the variables(without running the code). Also rewrite them using parenthesis to make the order explicit.

- Assume (x=0xFF33,MASK=0xFF00). Expression: c=x & MASK ==0;
- Assume (x=10,y=2,z=2;).Expression: z=y=x++ + ++y*2;
- Assume (x=10,y=4,z=1;).Expression: y>>= x&0x2 && z

Answers:

- c = 0. c = (x & (MASK == 0));
- z = 16, y = 16, x = 11. z=(y=((x++) + ((++y)*2)));
- x = 10, y =2, z = 1. y>>= ((x&0x2) && z)

Problem 2.5

Determine if the following statements have any errors. If so, highlight them and explain why.

- int 2nd_value=10;
- Assume (x=0,y=0,alliszero=1). alliszero =(x=1) && (y=0);
- Assume (x=10,y=3,z=0;). y=++x+y;z=z-->x;
- Assume that we want to test if last four bits of x are on. (int MASK=0xF;ison=x&MASK==MASK)

Answers:

- Error. Variable names should begin with '_' or characters, not digits.
- ~~No error.~~ '=' should be replaced with '=='. The correct version is `alliszero = (x==1) && (y==0);`
- ~~Error. z is integer, which doesn't have the method for `->`~~ ; `-->` may look suspicious, the expression simplifies to `y= (++x) + y; z = (z--) > x;`

- Error. The precedence order for `==` is higher than `&`. So the statement can't be used to test if last four bits of x are on.