Problem Set 2 – Solutions
Types, operators, expressions


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
Answer: On my 32-bit machine (/usr/include/limits.h,/usr/include/float.h), the sizes and limits are as follows. Results may differ if you used a 64 bit machine.

<table>
<thead>
<tr>
<th>Data type</th>
<th>size (bytes)</th>
<th>min</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>1</td>
<td>SCHAR_MIN (-128)</td>
<td>SCHAR_MAX (127)</td>
</tr>
<tr>
<td>unsigned char</td>
<td>1</td>
<td>0</td>
<td>UCHAR_MAX (255)</td>
</tr>
<tr>
<td>short</td>
<td>2</td>
<td>SHRT_MIN (-32768)</td>
<td>SHRT_MAX (32767)</td>
</tr>
<tr>
<td>int</td>
<td>4</td>
<td>INT_MIN (-2147483648)</td>
<td>INT_MAX (2147483647)</td>
</tr>
<tr>
<td>unsigned int</td>
<td>4</td>
<td>0</td>
<td>UINT_MAX (4294967295)</td>
</tr>
<tr>
<td>unsigned long</td>
<td>4</td>
<td>0</td>
<td>ULONG_MAX (4294967295)</td>
</tr>
<tr>
<td>float</td>
<td>4</td>
<td>FLT_MIN (1.175494e-38)</td>
<td>FLT_MAX (3.402823e+38)</td>
</tr>
</tbody>
</table>

Problem 2.2

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

- lower case letter (Answer: c>='a' && c<='z')
- upper case letter (Answer: c>='A' && c<='Z')
- digit (Answer: c>='0' && c<='9')
• white space (includes space, tab, new line) (Answer: c==’\n’| c==’\t’| c==’ ’)

Problem 2.3

Consider int val=0xCAFE; Write expressions using bitwise operators that do the following:
(a) test if at least three of last four bits (LSB) are on
(b) reverse the byte order (i.e., produce val=0xFECA)
(c) rotate four bits (i.e., produce val=0xECAF)

Answer:

(a) We have to test if last three or four bits are on. The possible values are 0x7,0xB,0xD,0xE,0xF.
   To test this, first we extract the last four bits. ( int bits=val&0xF; /*last four bits*/). Next we test if it is one of the possible patterns. ( bits==0x7 || bits==0xB || (bits>=0xD)).

(b) val = ((0xFF & val) << 8) | (val>>8)

(b) val = (val >> 4) | ((val&0xF)<<12)

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.

(a) Assume (x=0xFF33,MASK=0xFF00).Expression: c=x & MASK ==0;

(b) Assume (x=10,y=2,z=2;).Expression: z=y=x++ + ++y*2;

(c) Assume (x=10,y=4,z=1;).Expression: y>>= x&0x2 && z

Answer:

(a) The operator precedence is ’==’>’&’>’==’. Thus, the expression is equivalent to c= (x & (MASK==0)).
   Therefore x=0xFF33,c=0.

(b) The operator precedence is ’++’>’*’>’+’. Thus, the expression is equivalent to z = (x++) + ((++y)*2).
   Therefore x=11,y=3,z=10+3*2=16.

(b) The operator precedence is ’&’>’&’>’>>=’. Thus, the expression is equivalent to y>>= (x & 0x2) && z.
   Therefore x=10,y=2,z=1.

Problem 2.5

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

(a) int 2nd_value=10;

(b) Assume (x=0,y=0,alliszero=1). alliszero =(x=1) && (y=0);

(c) Assume (x=10,y=3,z=0;). y=++x+y;z=z-->x;

(d) Assume that we want to test if last four bits of x are on. (int MASK=0xF;ison=x&MASK==MASK)

Answer:

(a) Variable names cannot start with a number.

(b) ’=’ operator should be replaced with ’==’. The correct version is \( \text{alliszero } = (x==1) \&\& (y==0); \).

(c) There is nothing wrong with the statement. While \( -- > \) may look suspicious, the expression simplifies to \( y = (++x)+y; z = (z--) > x. \)

(c) There is nothing syntactically wrong with the statement. However, what we want is \( \text{ison } = (x \& \text{MASK}) == \text{MASK}. \) Based on operator precedence, the current expression simplifies to \( \text{ison } = x \& (\text{MASK} == \text{MASK}) \).

These exercises should have convinced you to use () always.