EE101 C programming and SW engineering 1
Lab Practice 3 – Basic Operations

Use your preferred compiler to investigate the programming exercises below. In each case, try to work out what the programs or sections of code will do **BEFORE** you compile and run the code. This will help you to test your understanding. Note Exercise 1 is a mathematics question; it does not require you to write any code.

Binary Numbers

Exercise 1

Choose four numbers between 0 and 255 and write them in the table below in binary and hexadecimal form.

Decimal number	Binary number	Hexadecimal

Operator Precedence, Associativity and Type Casting

Exercise 2

Assume all variables are of type int. Find the value of each of the following variables:

a) x = (12+6) / 2 * 3;

b) y = x = (2+3) / 4;

c) y = 3 + 2 * (x=7/2);

Exercise 3

Assume all variables are of type int. Find the value of each of the following variables:

a) x = (int) 3.8 + 3.3;

b) x = (2+3) * 10.5;

c) x = 22.0 * (int)3/10;

d) x = 22.0 * (int)(3/10);

Exercise 4

Write a working program that calculates each expression given in Exercise 2 and 3, in turn, and prints the result after each computation. Compile and run the program. Does the program produce the results you anticipated?

For example:

```
x = 3 * 4 + 2 / 3;                          /* compute  expression  */
printf("the result of the above is = %d" , x);   /* print expression  */
                                            /* compute next expression here, etc… */
```

Exercise 5

Try to figure out what values the following program will output; do the following:

- Write them down next to each executing line that generates an output.
- Compile and run the program.
- Write the values printed on the screen.
- Does the program produce the results you anticipated?

```
#include<stdio.h>

main()
{
int a = 1, b = 1, aplus, plusb;
aplus = a++;
plusb = ++b;
printf("a aplus plusb b \n");
printf("%1d \t%5d \t%5d \t%5d\n", a, aplus, plusb, b );
}
```

Note the numbers between % and the conversion character d in printf(). The number indicates the minimum field width (number of digits). For %f conversions it is possible to indicate the precision you want to display the floating-point number. For example:

```
printf("%3.5f",my_float);
/*will display 5 digits to the right of the decimal point of my_float.*/
```

The increment and decrement operators have a very high precedence of association. Only parentheses "( )" are higher. Therefore x*y++ means (x)*(y++). The increment and decrement operators affect a variable (unary operators) - not a combination of variables. Do not confuse precedence with the order of evaluation!

Exercise 6

Suppose you have the following:

```
y = 2;
n = 3
result = (y + n++) * 6;
```

What value does result get? Remember that the nature of the increment operator (postfix or prefix) determines when the value of n is changed!

Playing with strings

Exercise 7

Compile and run the following program. Write down what the program output will be.

```
#include <stdio.h>
#define PRAISE "you look great today :-)"

main()
{
char name[40];

printf("What is your name?\n");

scanf("%s", name);

printf("Hello, %s. %s\n", name, PRAISE);
}
```