Tutorial - Week 1

Objectives:
To discuss basic elements of C such as
- general structure of a C program
- variable declarations and data types
- input / output
- arithmetic expressions

1. Which of the following definitions are correct? If incorrect, give the reason:

```
a. int numStudents = 370;
b. int numLabs = 45.5;
c. double    1value, 2value;
d. char grade = 'P';
e. char stLevel = A;
f. char pinNum = 117;
g.      number = 24;
h. #define PI = 3.1416;
```

a. **correct**
b. Syntactically, it is correct. Logically, **it might be incorrect as you are assigning a real number to an integer variable, the fractional part will be removed but you won't get any error message**
c. **incorrect** (variable identifier (2value) starts with a number)
d. **correct**
e. **incorrect** (the character is missing the single quotation, unless A has already been declared as a char
f. **correct**
g. **incorrect** (no data type)
h. **incorrect** (there should be no equal sign and semi colon at the end)

2. Assuming the following variable declarations, what output is produced by `printf()`?

```
int numOfModules = 15, moduleCode = 7;
float width = 65.243, height=38.159;
char secLevel = 'B';
```

```
a. printf("Number of Modules=%6d\n",  numOfModules);
```

Displayed Output:

```
Number of Modules= ~~~~15
```

```
b. printf("Module Code = %04d\n",  moduleCode);
```

Displayed Output:

```
Module Code = 0007
```

```
c. printf("Width = %-8.2fHeight = %-8.2f\n", width,  height);
```

Displayed Output: `Width = 65.24~~~Height = 38.16~~~`

d. `printf("Dimensions = %.1f x %.1f mm\n", width,  height);`

Displayed Output:
```
Dimensions = 65.2 x 38.2 mm
```

e. `printf("Security Level:%3c\n",  secLevel);`

Displayed Output:
```
~~B
```

3. Is `scanf()` used correctly?

```
float x, y;

a. scanf("Enter a number: %f", &x);
b. scanf( "%d" , &x);
c. scanf( "%3f", &x);
d. scanf( "%f, %f", &x, &y);
```

Please note that the above statements will compile. However, an incorrect scanf text format leads to incorrect read operation and hence variable value.

a. Incorrect use: quoted text in `scanf` first parameter is not expected.
b. Incorrect as `x` is a float, not an integer
c. Only 3 digits, including the decimal point, will be read from the user's input.
d. Incorrect. No comma should be used between the format specifiers in `scanf`.

4. Correct mistakes in the program that converts a temperature in Fahrenheit to Celsius: Celsius = 0.55*(fahrenheit – 32)

```
#include <stdio.h>
#define   COEFF = 0.55
int main(void)
{
     float celsTemp, fahrTemp

     printf(' Enter the temperature in Fahrenheits: ');
     scanf("%f",    fahrTemp );
     celsTemp = COEFF *     fahrTemp – 32;
     printf("Celsius temperature = %.1f" , fahrtemp  );

     return 0;
}
```

- `#define COEFF = 0.55 should be #define COEFF 0.55 since COEFF is a symbol name`
- `float celsTemp, fahrTemp should be float celsTemp, fahrTemp;`
- `printf(' Enter the temperature in Fahrenheits: '); should be printf(" Enter the temperature in Fahrenheits:");`
- `scanf("%f",fahrTemp ); should be scanf("%f",   &fahrTemp );`

- `celsTemp = COEFF * fahrTemp - 32;` should be
  `celsTemp = COEFF *(fahrTemp - 32);`
- `printf("Celsius temperature = %.1f" , fahrtemp );` should be
  `printf("Celsius temperature = %.1f" , celsTemp );`

5. Which of the following declarations are correct? If incorrect, give the reason:

a. `char productType = 'V';`
b. `char minutes = 45;`
c. `char days =172;`
d. `char value = -15;`
e. `float width = 32.157e2;`
f. `int length = width;`

Syntactically, the above statements are correct. Logically, the statement

a. is correct
b. is correct
c. leads to an overflow as char values range is between -128 and 127
d. is correct
e. is correct
f. demotes the width value from float to integer, i.e. its fractional part will be truncated.

6. What value will be assigned to the variables (taking into account all the preceding operations)

```
int intRes, number = 5;
float fpRes;
char      grade = 'A';
```

a. `intRes = 10/3;`
b. `intRes = 10%3;`
c. `intRes = 15/2/3;`
d. `fpRes = 15.0/2;`
e. `intRes = number++;`
f. `intRes = ++number;`
g. `intRes = (7 + 3)/2;`
h. `intRes += 4;`
i. `fpRes = intRes/2;`
j. `fpRes = (float)intRes/2;`
k. `grade += 2;`

a. 3 : result of integer division as both operands are integers
b. 1: remainder of the division of the operands
c. 2 : parenthesized version of the expression is ((15/2) /3)= (7/3)=2
d. 7.5 : real division since one operand is not a whole number
e. 5 : post-incrementation operator, the current value of number is first assigned to intRes before being incremented. The resulting values of intRes and number are 5 and 6,

respectively.

f. 7 : pre-incrementation operator, the variable number is first incremented before being assigned to intRes. The resulting value of ntRes and number 7 and 7.

g. 5 as the integer division of 10 by 2

h. 9 : compound operator "+=", the lvalue (intRes) is incremented by the rvalue (4)

i. 4.0 : the integer division is promoted to float. The magnitude remains the same.

j. 4.5 : As we are casting the nominator to the float, the division execution will be real division and no longer integer

k. 'C' : use of the compound operator "+="; the addition is on a character variable; an increment by 2 on an alphabet letter returns the alphabet letter at offset 2 from the current one

7. Evaluate the following:

```
float fa = 29.0, fb = 10.0, fc = 10.37;
int ia;

a. ia = (int)(fa/fb);
b. ia = fa/fb;
c. How to round a float to the nearest int?
d. How to assign only the integer part of fc to fa?


a. ia = 2; casting the real division result (2.9) to integer
b. ia = 2; converting the real division result(2.9) to integer (removing
   the fractional part)
c. (int) (fa + 0.5) if fa is positive and (int) (fa - 0.5) if it is
   negative. Otherwise, use the function round()
d. fa = (int) fc;
```

8. Specify the `lvalue` and `rvalue` in the following statements?

```
int a, b=1;

a. a = 1;
b. ++a;
c. b = b + a;
d. a = b++;
e. b = 10++;
f. b = ++(1-a);
```

a. The value *1* is a `rvalue`, whereas the variable *a* is a `lvalue`

b. The variable *a* is a `lvalue`

c. The expression b + a is a `rvalue`, whereas *b* is an `lvalue`

d. a copy of *b* becomes a `rvalue` and assigned to the `lvalue` *a*, then the `lvalue` b is incremented

e. Error: the post incrementation/decrementation can only be applied on a variable, 10 is not a variable

f. Error: the post incrementation/decrementation can only be applied on a variable, (1-a) is an expression and not a variable

9. Write the `#define` preprocessor directive and declarations for a program that has a

constant macro for `PI` (3.14159) and variables `radius, area, and circumf` declared as `double`, variable `num_circ` as an `int`, and variable `circ_name` as a char.

```
#include <stdio.h>
#define PI 3.14159

int
main(void)
{
   double  radius, area, circumf;
   int     num_circ;
   char    circ_name;
   /* executable statements omitted */
 }
```

10. Write a statement that displays the following line with the value of the type `int` variable `n` before the period.

```
The value of n is ___________.

printf("The value of n is %d.\n", n);
```

11. Assuming that `side` and `area` are type `double` variables containing the length of one side in cm and the area of a square in square cm, write a statement that will display this information in this form:

```
The area of a square whose side length is ______ cm is ______ square cm.

printf("The area of a square whose side length is %lf cm is %lf square
cm.\n", side, area);

The l modifier is required in scanf with double, but not in printf.
```

12. Show how the value −3.6175 would be printed using the formats `%8.4f, %8.3f, %8.2f, %8.1f, %8.0f, %.2f.`

```
value = -3.6175    (# means blank)

  Format     Output
  %8.4f     #-3.6175
  %8.3f     ##-3.618
  %8.2f     ###-3.62
  %8.1f     ####-3.6
  %8.0f     #####-4.
  %.2f      -3.62
```