The cast and sizeof Operator

```
Word definitions:
Truncated =shorten (something) by cutting off the top or the end.
Implicit =essentially or very closely connected with; always to be found in.
```

Type Conversions

- Conversion of data between types can happen automatically (implicit conversion) by the language or explicit by the program.
  - ➔ To effectively develop C programs, you must understand the rules used for the implicit conversion of floating-point and integer values in C.
- Normally, you shouldn't mix types, but there are occasions when it is useful
  - ➔ Remember, c is flexible, gives you the freedom, but do not abuse it.
- Whenever a floating-point value is assigned to an integer variable C , the decimal portion of the number gets truncated.

```
int x = 0;
float f = 12.125;
x = f; // value stored in x is the number 12, only the int portion is stored
```

- Assigning an integer variable to a floating variable does not cause any change in the value of the number.
  - ➔ Value is converted by the system and stored in the floating variable.
- When performing integer arithmetic
  - ➔ If two operands in an expression are integers then any decimal portion resulting from a division operation is discarded, even if the result is assigned to a floating variable.
  - ➔ If one operand is an int and the other is a float the the operation is performed as a floating point operation.

The Cast Operator

- As mentioned, you should usually steer clear of automatic type conversions, especially of demotions.
  - ➔ Better to do an explicit conversion,
- It is possible for you to demand the precise type conversion that you want .
  - ➔ Called a cast and consists of preceding the quantity with the name of the desired type in parentheses.
  - ➔ Parentheses and type name together constitutes a cast operator, i.e (type)
  - ➔ The actual type desired, such as long , is substituted for the word type.

- The type cast operator has a higher precedence than all the arithmetic operators except the unary and minus and unary plus.

```
(int) 21.51 + (int) 26.99
```

- Is evaluated in C as

```
21 + 26
```

The sizeof Operator

- You can find out how many bytes are occupied in memory by a given type by using the sizeof operator.
  - ➔ Sizeof is a special keyword in C
- Sizeof is actually an operator, and not a function
  - ➔ Evaluated at compile time and not at runtime, unless a variable-length array is used in its arguments.
  - ➔ The argument to the sizeof operator can be a variable, an array name, the name of a basic data type , the name of a derived data type, or an expression.
- sizeof(int) will result in the number of bytes occupied by a variable of type int
- You can also apply the sizeof operator to an expression,
  - ➔ Result is the size of the value that results from evaluating the expression.
- Use the sizeof operator wherever possible to avoid having to calculate and hard-code sizes into your program

Other Operators.

- The asterisk “*” is an operator that represents a pointer to a variable.
  *a;
- ?: is an operator used for comparisons.
- If Conditional is true ? then values X : otherwise value Y
- Name is the ternary operator