Data Type & Operators

Dr. Hyun Lim

h.lim@lboro.ac.uk

Institute for Digital Technologies

Loughborough University London

Basic Data Types

- Basic built-in datatypes:
  - int: **int**eger numbers, *e.g.* int a = 5;
  - float: **float**ing point numbers, *e.g.* float b = 6.2;
  - double: big floating point numbers, twice the storage for the number. 8 bytes on PCs

    e.g. double c = 1.0/3; 1./3; 1/3.0; 1/3.

    double c_a = 1/3;   // in this case c_a == 0
  - char: **char**acters, *e.g.* char d = 's';
  - bool: depends on the compiler, for example *NewbieIDE* provides a bool type. A **bool**ean (*true*/*false*) data type: 0 is ***false***, otherwise ***true*** in C.

Exercise

```
int main()
{
    int nr=1;
    char name ='A';
    printf("Display book information\n");
    printf("No.[%i]\n", nr);          // %i integer number
    printf("Author: %c\n", name);  // %c character
        // getchar();  // Optional
        // %i %c: format specifiers
return 0;
}
```

Character and (Character) String

```
int main()
 {
  char index = 'a';     // Character
  printf("\n %c \n ", index);

  char author[] = "Dennis Ritchie"; //(Character) String
  printf("\n %s \n ", author);
                            // cf. [Python] print("%s" %bb)
  return 0;
 }
```

Variables – General recommendations

- **Avoid** using **global** variables, but **local**, *i.e.* do not define global variables unless it is really necessary.
- Use **const** for **constants**, which is a *type qualifier*.
- Not to use ~~goto~~ in C (an unconditional jump to a labelled statement)

```
~~int nr=1;~~
~~float aa=3.2;~~                      global variables

int main()
   {  int my_index;
      int my_book_nr = 0; // initialisation
     const int max_book_nr = 100;           local variables

     return 0;
   }
```

Operators

Arithmetic Operators

- C has a number of arithmetic operators which are used to combine variables and constants into expressions
  - Unary operators +, -

    : e.g. +x, -x
  - Binary operators +, -, *, /

    : e.g. x+y*z
- **Int**eger expression examples
  - 3*4 = 12, 17/5 = 3

    Note that integer division discards any fractional part.
- Examples for **floating point numbers**
  - 3*4 = 12, 3.0/4, 3/4.0
  - However, float aa = 3/4, aa == 0

Assignment Operators

- **Assignment** operator **=**

  e.g. x=y, *cf.* x==y (== is a relational operator)

- Other useful shorthand:

  a+=b means a=a+b

  a-=b means a=a-b

  a*=b means a=a*b

  a/=b means a=a/b

  a%=b means a=a%b // % here is the modulus operator

- *a*++ means "increased by 1"

Relational Operators

- They are >, <, >=, <=, ==, !=
- Expressions with relational operators **evaluate to true / false**,
  - e.g.

    27>21 // true

    27<=3 // false
- In C there is originally no ***bool**ean* data type, but integers can be used, instead:
  - **0: false, anything else: true**
  - **It depends on the compiler**

Logical Operators

- && (logical ***and***): e.g. A && B
- || (logical ***or***): e.g. A || B
- ! (logical ***not***): e.g. !A

Exercise

```
 int index= 1;    //Try to put 2 or any other numbers, instead
 int number = 20;     // Try to put 50, instead
 const int max_nr = 50;

 if (index == 1 && number < max_nr)
   { printf("Insert the book information. \n \n \n");
     number++;     // increase the input number
   }
else if (index == 1 && number >= max_nr)
                          //if more than the max. number
     printf("You cannot put any more item. \n \n \n");
 else if (index == 2) // Display the data set on a screen
     printf("Index(%i) will display the book title. \n \n \n ", index);
 else // In choosing a wrong index nr.
     printf("Mistake! Please start it again. \n \n \n");
  // while (!_kbhit()); // It's optional
```