COMP1917: 01_Numbers In, Numbers Out

Sim Mautner

simm@cse.unsw.edu.au

July 26, 2016
References

- Moffat, Chapter 2.
Variables and Types

- Variables are used to store data. \(\rightarrow\) boxes
- Each variable has a type. \(\rightarrow\) size/structure of the box
- For now, we are using 3 data types:
  - `char` character ‘A’, ‘e’, ‘#’
  - `int` integer 2, 17, -5
  - `float` floating point number 3.14159, 2.71828
Variables

- **Declare**
  The first time a variable is mentioned, we need to specify its type.

- **Initialise**
  Before using a variable we need to assign it a value.

- **Assign**
  To give a variable a value.

```c
int num;   // Declare
num = 5;   // Initialise (also Assign)
...        // Other code
num = 27;  // Assign
```
Variables

- We can also Declare and Initialise in the same step:

```java
int num = 5; // Declare and Initialise
...
num = 27; // Assign
```
Variable Names (and other Identifiers)

- Must be made up of letters, digits and underscores (‘_’)
- The first character must be a letter
- Are case sensitive (num1 and Num1 are different)
- **Restrictions:** Keywords like:
  
  ```
  if, while, do, int, char, float
  ```

  cannot be used as identifiers
Output using printf()

- No variables:
  
  ```c
  printf("Hello World\n");
  ```

- A single variable:
  
  ```c
  int num = 5;
  printf("num is %d\n", num);
  ```

- More than one variable:
  
  ```c
  int num1 = 5;
  int num2 = 17;
  printf("num1 is %d and num2 is %d\n", num1, num2);
  ```

**Note:** The order in which the variables are listed, is the order in which they will appear.

```c
int num1 = 5;
int num2 = 17;
printf("num2 is %d and num1 is %d\n", num2, num1);
```
Output using printf()

Placeholders:

- char uses %c
- int uses %d
- float uses %f
- double uses %lf

Try It Yourself:

1. Copy the code from the end of the previous slide into a C program and run it.

2. Make the appropriate changes so that it declares, initialises and prints a char, float and double.
Input using `scanf()`

- Example:
  ```c
  int num = 0;
  scanf("%d\n", &num);
  printf("num = %d\n", num);
  ```

- Notice that the variable is still initialised. (Not necessary, but good practice.)

- Notice the `&` before the variable name. Don’t forget it!!
Input using `scanf()`

- **Multiple variables (space separated):**
  ```c
  int num1 = 0;
  int num2 = 0;
  scanf("%d %d\n", &num1, &num2);
  printf("num1 = %d and num2 = %d\n", num1, num2);
  ```

- **Multiple variables (comma separated):**
  ```c
  int num1 = 0;
  int num2 = 0;
  scanf("%d, %d\n", &num1, &num2);
  printf("num1 = %d and num2 = %d\n", num1, num2);
  ```

- Notice the space or comma between the variables.
Input using `scanf()`

Try It Yourself:

- Create a C program using the code from the previous slide.
- Using what you know about placeholders for `printf()` (earlier this lecture) and `scanf()`, make the changes required so that it scans in and prints out a character (`char`).
Programming Task

Write a program to:

1. Read in a number.
2. Compute the cube of that integer.
3. Display the result on the screen.
Programming Task

Process:

1. Step 1: Think about the problem.
2. Step 2: Break it down into steps (and each step into smaller steps).
3. Step 3: Convert the basic steps into code.
4. Step 4: Compile the program.
5. Step 5: Test the program on a range of data.
## Arithmetic Operators

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Example</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>+</td>
<td>a + b</td>
<td>none</td>
</tr>
<tr>
<td>Subtract</td>
<td>-</td>
<td>a - b</td>
<td>none</td>
</tr>
<tr>
<td>Multiply</td>
<td>*</td>
<td>a * b</td>
<td>none</td>
</tr>
<tr>
<td>Divide</td>
<td>/</td>
<td>a / b</td>
<td>ignores remainder for integer division</td>
</tr>
<tr>
<td>Modulus</td>
<td>%</td>
<td>a % b</td>
<td>remainder of a/b</td>
</tr>
</tbody>
</table>

```c
int result = 50 / 3;
int remainder = 50 % 3;
printf("50 divided by 3 equals %d remainder %d\n", result, remainder);
```
printf() with floats

%d    decimal integer
%5d   decimal integer at least 5 chars wide
%f    floating point number
%5f   floating point number at least 5 chars wide
%.3f  floating point number 3 decimal places
%5.3f floating point number at least 5 chars 3 decimal places
Math Equations

- `sqrt()`, `sin()`, `cos()`, `log()`, `exp()`
- `#include <math.h>`
- Compile with `-lm`
#include <stdio.h>
#include <math.h>

int main(int argc, char *argv[]) {

int num = 0;
scanf("%d", num);
double result = sqrt(num);
printf("The square root is: %.3lf\n");
return 0;
}