General Form of a Program

Preprocessor directives

**main function heading**

```
{
  declarations
  executable statements
  return statement
}
```

Programming Style

- **Usage of Comments**
  - Always label your program
    ````
    /*
    * Programmer : John Smith
    * Date First Completed : 10.10.2010
    * Date Last Modified : 11.11.2010
    *
    * Calculates the average of number provided as a list
    */
    ```
  - Always label your variable definitions
    ````
    double pounds, /* input – weight in pounds */
    kgs; /* output – weight in kilograms */
    ```
  - Always label major steps in your algorithm
    ````
    /* Convert weight into kilograms*/
    ```
Programming Style

- **Usage of spaces**
  - $z = x + y \rightarrow z = x + y$
  - `int k, l, m; \rightarrow int k, l, m;`

- **Variable Declaration**
  - Always declare variables at the beginning, right after function heading

- **Indentation**
  - Use tabs to make your program more readable

```c
#include <stdio.h>
define KGS_PER_LBS 0.454

int main (void)
{
    double pounds, kgs;
    /* Get weight in Pounds */
    printf("Enter the weight in Pounds: ");
    scanf("%lf", &pounds);
    /* Convert weight into kilograms*/
    kgs = KGS_PER_LBS * pounds;
    /* Display weight in kilograms*/
    printf("The weight in kilograms is \%.2f, kgs\n");
    return(0);
}
```

Parts of a Program

- /* Converts Pounds into Kilograms */
  - /* Library Definitions */
  - /* Conversion Constant */

- #include <stdio.h>
  - #define KGS_PER_LBS 0.454

- int main (void)
  - { double pounds, kgs;
  -     /* Get weight in Pounds */
  -     printf("Enter the weight in Pounds: ");
  -     scanf("%lf", &pounds);
  -     /* Convert weight into kilograms*/
  -     kgs = KGS_PER_LBS * pounds;
  -     /* Display weight in kilograms*/
  -     printf("The weight in kilograms is \%.2f, kgs\n");
  -     return(0);
  - }

- /* Library Definitions */
  - /* Conversion Constant */

- int main (void)
  - { double pounds, kgs;
  -     /* Get weight in Pounds */
  -     printf("Enter the weight in Pounds: ");
  -     scanf("%lf", &pounds);
  -     /* Convert weight into kilograms*/
  -     kgs = KGS_PER_LBS * pounds;
  -     /* Display weight in kilograms*/
  -     printf("The weight in kilograms is \%.2f, kgs\n");
  -     return(0);
  - }
C Language Elements

- Comments
  - Enclosed in /* ... */
- Preprocessor Directives
  - Starts with #
- Constants
  - #define directive
- Reserved Words
  - void, main, int, double, return, ...

C Language Elements

- Standard Identifiers
  - Defined in libraries, header files,
  - printf, scanf, ...
- Variables (User-Defined Identifiers)
  - Contains only letters, digits, and underscores
  - Cannot start with a digit
  - No reserved words can be used
  - Not recommended to use Standard Identifiers
  - Uppercase, Lowercase letters are considered different
    - Size, size, and SIZE are considered to be different definitions
    - monthlysalary, monthly_salary, or MonthlySalary
Data Types

- Declaration Syntax
  - Type variable_list;
  - Type can be int, double, or char
  - variable_list is a comma separated list
  - ; ends the declaration

- Example
  ```
  float kms, miles;
  double kms, miles;
  char FirstInitial, SecondInitial;
  int age;
  ```

Data Types

- **Type int**
  - Represents positive and negative integers

- **Type float**
  - Represents positive and negative floating point numbers (real numbers)
  - Has both integer and fractional parts

- **Type double**
  - Represents big floating point numbers
  - Scientific number notation can be used for very large and very small numbers
    - 1234000000.0 = 1.234e09
    - 0.000001234 = 1.234e-06
Data Types

- **Type** `char`
  - Represents a single character value
  - It can be letter, digit, or a special symbol
  - The character should be enclosed in single quotes ‘c’
  - Example
    
    ```
    char asterix;
    asterix = 'a';
    ```

<table>
<thead>
<tr>
<th>Type</th>
<th>Version</th>
<th>Length (Bytes)</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>char</td>
<td>1</td>
<td>256 chars</td>
</tr>
<tr>
<td>int</td>
<td>short</td>
<td>2</td>
<td>-215 to 215-1</td>
</tr>
<tr>
<td>int</td>
<td></td>
<td>4</td>
<td>-231 to 231-1</td>
</tr>
<tr>
<td>long</td>
<td></td>
<td>4</td>
<td>-231-1 to 231-1</td>
</tr>
<tr>
<td>unsigned short</td>
<td>2</td>
<td>0 to 215-1</td>
<td></td>
</tr>
<tr>
<td>unsigned</td>
<td></td>
<td>4</td>
<td>0 to 232-1</td>
</tr>
<tr>
<td>unsigned long</td>
<td></td>
<td>4</td>
<td>0 to 252-1</td>
</tr>
<tr>
<td>float</td>
<td>float</td>
<td>4</td>
<td>10^-37 to 10^38</td>
</tr>
<tr>
<td>double</td>
<td></td>
<td>8</td>
<td>10^-307 to 10^308</td>
</tr>
<tr>
<td>long double</td>
<td></td>
<td>12</td>
<td>10^-4931 to 10^4932</td>
</tr>
</tbody>
</table>