**Preprocessor definitions.**

#define

Another mechanism to name constant values is the use of preprocessor definitions. They have the following form:

*#define identifier replacement*

After this directive, any occurrence of identifier in the code is interpreted as replacement, where replacement is any sequence of characters (until the end of the line). This replacement is performed by the preprocessor, and happens before the program is compiled, thus causing a sort of blind replacement: the validity of the types or syntax involved is not checked in any way.

For example:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 | #include <iostream>  using namespace std;  #define PI 3.14159  #define NEWLINE '\n'  int main ()  {  double r = 5.0; // radius  double circle;  circle = 2 \* PI \* r;  cout << circle;  cout << NEWLINE;  } | 31.4159 | [Edit & Run](https://cplusplus.com/doc/tutorial/constants/) |

Note that the #define lines are preprocessor directives, and as such are single-line instructions that -unlike C++ statements- do not require semicolons (;) at the end; the directive extends automatically until the end of the line. If a semicolon is included in the line, it is part of the replacement sequence and is also included in all replaced occurrences.