Cuando se trabaja con teoría de computadores, de lenguajes de programación y de otros sistemas lógicos, es común toparse con el término Turing completo, que es el que tiene un poder computacional equivalente a la máquina de Turing universal. Esto quiere decir que el sistema y la máquina universal de Turing pueden emularse entre sí.

Aunque físicamente es imposible que estas máquina existan, debido a que requieren un almacenamiento ilimitado y probabilidad de cero falla, la completitud de Turing es atribuida a máquinas físicas o lenguajes de programación que podrían ser universales si contaran con almacenamiento infinito y totalmente fiables.

La primera de esas máquinas apareció en 1941, tratándose de la Z3, de Konrad Zuse, y era controlada por programas. Su universalidad se demostró mucho tiempo después, por Raúl Rojas, en 1998.

Hablar de Turing completo es significativo, ya que cada diseño verosímil de un dispositivo de computación, por más avanzado que sea, pueden ser emuladas por una máquina universal de Turing. De este modo, una máquina que pueda actuar como una universal de Turing, puede hacer cualquier cálculo que cualquier otra computadora sea capaz de hacer.

También existe la hipótesis de que el Universo es Turing completo, más detallado en las implicaciones filosóficas en la Tesis de Church-Turing y en Física digital.

Un ejemplo de Turing completo son las expresiones regulares que tienen algunos lenguajes como Perl, otro podría ser el lenguaje de macros de Excel.

Hablando de teoría de computabilidad, existe un resultado que concluye que es imposible saber si un programa escrito en un lenguaje Turing completo se continuará ejecutando indefinidamente o se detendrá en un periodo finito de tiempo. Un método para prevenir que suceda lo primero, es hacer que los programas se detengan luego de un tiempo determinado, sin embargo, esos sistemas ya no serían Turing completos.

El cálculo lambda sin tipo es Turing completo, pero muchos cálculos lambda con tipo, incluyendo el Sistema F no lo son.