Signal Processing (Signal Handling) in C ++

Signal (Signal) is the interrupt that is distributed to an operating system process that can end a program. You can create interrupts by pressing CTRL + C on UNIX systems, LINUX, Mac OS or Windows.

Signal (Signal) is the interrupt that is distributed to an operating system process that can end a program. You can create interrupts by pressing CTRL + C on UNIX systems, LINUX, Mac OS or Windows.

There are signals that cannot be captured by the program, but there are also signals that you can capture in your program, and can perform appropriate actions based on that signal. These signals are defined in the Header file of C ++.

SignalDescriptionSIGABRT Abnormal termination of the program, for example a call to **abort** SIGFPE An arithmetic operation is not correct, such as dividing by zero or an overflow operation SIGILL Detecting a thread SIGINT invalid command Receiving an SIGSEGV interactive signal An invalid access to storage SIGTERM An end request is sent to the program

Function signal () in C ++

The signal processing library in C ++ provides a signal function to trap unexpected events. Here is the syntax of the signal () function in C ++:

```
void (*signal (int sig, void (*func)(int)))(int);
```

This function takes two parameters: the first parameter is an integer that represents the signal number and the second parameter is a pointer to the signal processing function.

Now, write a simple C ++ program to catch the SIGINT signal using the signal () function in C ++. Whatever signal you want to catch in the program, you must write that signal using the signal function and link it to a Signal Handler. You consider the example:

```
#include #include using namespace std; void signalHandler( int tinhieuso ) { cou
```

Compiling and running the above C ++ program will produce the following results:

Now, press CTRL + C to interrupt the program and you will see that the program will catch this signal and will print something like this:

```
Going to sleep. Going to sleep. Going to sleep. Tin hieu ngung chuong trinh (2) c
```

Function raise () in C ++

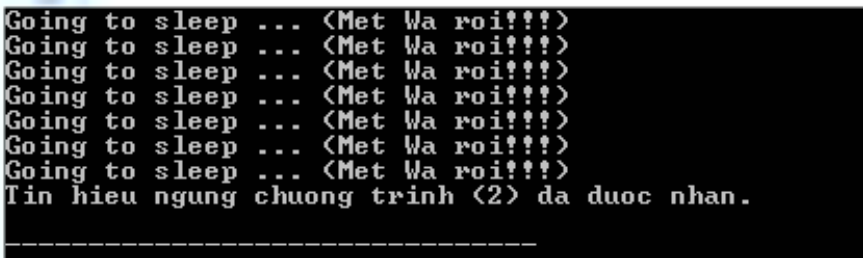
You can generate signals by the **raise ()** function in C ++, but receive an integer that represents the signal number as a parameter and has the following syntax:

```
int raise (signal sig);
```

Here, the **sig** is the signal number to send any type of signal: SIGINT, SIGABRT, SIGFPE, SIGILL, SIGSEGV, SIGTERM, SIGHUP. Here is an example of creating an internal signal using the raise () function in C ++, as follows:

```
#include #include using namespace std; void signalHandler( int tinhieuso ) { cou
```

Compiling and running the above C ++ program will produce the following results:



According to Tutorialspoint

Previous article: Preprocessor in C ++

Next article: Multithread in C ++