C library function - raise()

Description

The C library function **int raise(int sig)** causes signal **sig** to be generated. The **sig** argument is compatible with the SIG macros.

Declaration

Following is the declaration for signal() function.

```
int raise(int sig)
```

Parameters

- **sig** − This is the signal number to send. Following are few important standard signal constants −

Sr.No.	Macro & Signal
1	**SIGABRT** (Signal Abort) Abnormal termination, such as is initiated by the abort function.
2	**SIGFPE** (Signal Floating-Point Exception) Erroneous arithmetic operation, such as zero divide or an operation resulting in overflow (not necessarily with a floating-point operation).
3	**SIGILL** (Signal Illegal Instruction) Invalid function image, such as an illegal instruction. This is generally due to a corruption in the code or to an attempt to execute data.
4	**SIGINT** (Signal Interrupt) Interactive attention signal. Generally generated by the application user.
5	**SIGSEGV** (Signal Segmentation Violation) Invalid access to storage − When a program tries to read or write outside the memory it is allocated for it.
6	**SIGTERM** (Signal Terminate) Termination request sent to program.

Return Value

This function returns zero if successful, and non-zero otherwise.

Example

The following example shows the usage of signal() function.

```
#include <signal.h>
#include <stdio.h>

void signal_catchfunc(int);

int main () {
   int ret;
```

```
   ret = signal(SIGINT, signal_catchfunc);

   if( ret == SIG_ERR) {
      printf("Error: unable to set signal handler.\n");
      exit(0);
   }
   printf("Going to raise a signal\n");
   ret = raise(SIGINT);

   if( ret !=0 ) {
      printf("Error: unable to raise SIGINT signal.\n");
      exit(0);
   }

   printf("Exiting...\n");
   return(0);
}

void signal_catchfunc(int signal) {
   printf("!! signal caught !!\n");
}
```

Let us compile and run the above program to will produce the following result −

```
Going to raise a signal
!! signal caught !!
Exiting...
```