To help with the challenges introduced in workshop 1, we offer the following information to use execvp().

[1] int execvp(const char \*file, char \*const argv[]);

1. The first input is the char\*file, which is the filename of the other executable that we are going to call. The data type is the character pointer, which is pointed to a string.

Note that when setting \*file, we need to also include the path of the executable, if the executable is not a system command.

Note that we could only call the executable instead of the source file.

1. In addition to \*file, we also have the char \* const argv[] input in execvp().

\* argv[] is an array of pointers to null-terminated strings. \*argv[] represent the argument list given to the new program (the executable called counting in our workshop).

There are a couple of things that we need to pay special attention.

<1> Each argv[i] is a pointer to a null-terminated string.

<2> argv[0], by convention, needs to be the filename to be executed. This means that we will have the executable filename in both *file* and *argv*[0].

[2] The following figure shows the connection between execvp() and the counting program

Text

Description automatically generated

Note that char\*\* argv is equivalent as char\* argv[].

Note that argc is *automatically* calculated from the size of char\* argv[]. We don’t need to set it.

[3] type conversion in c++

There are many ways for converting between an integer and a string in C. The following is just one example.

1. To convert a string to an integer, we could use the atoi() function
2. To convert an integer to a string, there are multiple ways

<1> itoa(). However, the itoa() is not a standard function and might be missing on your platform.

<2> sprintf().

Taking the following short code for example.

char stringNum[20];

int num=100;

sprintf(stringNum,"%d",num);

<3> You could also use to\_string() or stringsteam in c++