

Command Line Arguments



Systems Programming

Command Line Arguments

- A C program must always have a function named **main**. This function is directly invoked by the Linux/Unix system.
- **main** has two arguments conventionally named **argc** and **argv**.
- The **argc** argument is of type **int** and corresponds to the number of arguments provided on the command line (including the program name as the first argument).

Command Line Arguments

- The second argument to main, **argv**, is an array of pointers to strings.
- Each string contains the ASCII string representation of what is typed on the program command line.

Command Line Arguments

For example, if the command line typed is:

```
./prog3 file1 200
```

argc will have the value **3**

and

argv[0] **points** to string **"./prog3"**

argv[1] **points** to string **"file1"**

argv[2] **points** to string **"200"**.

Command Line Arguments

- It is standard and a **safe programming** practice for main to immediately check to see if it has received the correct number of arguments from the Unix command line.
- If there is a mismatch, main prints out a **proper usage statement** and immediately ends the program.

Command Line Arguments

- For command line arguments intended as integer parameters to the program, the ASCII string representation of that integer has to be converted to an integer using the standard library function `atoi.{ascii-to-integer}`
- See pages 333-334 in D&D for complete syntax and an example of `atoi` usage.

A Command Line Argument Sample Program

```
/* An Example of the Use of Command Line Arguments */
```

```
#include <stdio.h>  
#include <stdlib.h>  
#define SIZE 100
```

Note: main function arguments

```
int main (int argc, char *argv[])  
{
```

```
    int i, samples, table[SIZE];
```

```
    char *samstring, *timestring; /* strings are needed */
```

```
    char *progstring;
```

```
    if(argc != 3)
```

Check number of arguments

```
        printf("Proper Usage is: com-arg samples time\n");
```

A Command Line Argument Sample Program

```
else
{
    progstring = argv[0];
    samstring = argv[1];
    timestring = argv[2];
    printf("Program = %s\n", progstring);
    samples = atoi(samstring);    /* convert */
    printf("Please enter %d samples\n", samples);
    for (i=0; i < samples; i++)
        scanf("%d", &table[i]);
    for (i=0; i < samples; i++)
        printf("sample[%d] = %d\n", i+1, table[i]);
    printf("Time = %d\n", atoi(timestring));
}
return;
}
```

```
./com-arg 3 500
Program = ./com-arg
Please enter 3 samples
745
1023495
2
sample[1] = 745
sample[2] = 1023495
sample[3] = 2
Time = 500
```


Command Line Arguments

- Using command line arguments to specify variable size for input parameters is a preferred technique.
- Remember *C* treats the arguments as ASCII strings.
- Use **atoi** to convert the strings to integers.