Command Line Arguments

Log into your Unix account and create a directory called csc2405 in your home directory. Change your working directory to **csc2405** (recall the Unix command **cd csc2405**). In the directory **csc2405**, create another directory called **mainargs**, then change your working directory to **mainargs**. Complete the three exercises below, making sure you understand the use of command line arguments in your program.

Exercise 1

C, C++ and Java allow you to input values into your program through the command line. You will only need to declare the **main** function in a slightly different form. In C (and C++), this is

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

- **argc** is the number of command line arguments (including the name of the executable)
- **argv** is an array of the arguments

Type the following program in a file called **mainargs.c**:

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

    printf("\nYour program name is %s\n", argv[0]);
    printf("Your arguments are: \n");
    for(i = 1; i < argc; i++)
       printf("\t \t %s starts with %c \n",
                    argv[i], argv[i][0]);
    return 0;
}
```

Create an executable called **mainargs** and run it using the command

```
./mainargs let us see what this does
```

Inspect the output of your code and analyze the code to answer the following questions?

1. What data type is **argc**?
2. What value does **argc** take when the program gets invoked?
3. What data type is **argv**?
4. What data type is **argv[0]**?
5. What gets stored in **argv[0]**?
6. What values are stored in **argv** when the program gets invoked? Draw a diagram illustrating the memory space occupied by **argv**.

Exercise 2

Use the manual pages on the Sun system to learn about the C function **atoi**. To do so, type in the command

```
man atoi
```

You will find out that the **atoi** function takes a string (`char *` in C) as an argument, and converts it to an integer. For instance, in the C statement

```
x = atoi("123")
```

the **atoi** function takes as argument the ***string*** **"123"** and returns the ***integer*** value **123** (one hundred and twenty three) in the **int** variable x.

Exercise 3

Rewrite the program from Exercise 2 to get the two numbers from the command line and print out their sum. Save the program into a file called **sumargs.c** and run it using a command similar to the following:

```
./sumargs 7 11
```

Have your program print an error message in case the command line is invalid:

```
if(argc < 3)
{
   printf("Invalid command line: please supply two integers\n");
  exit(1);
}
```

Note that you'll need to use the **atoi** function to convert the arguments (which are strings) to integer values.

What to turn in:

Call the instructor if finished by the end of this class. Otherwise, at the beginning of next class, hand in a printout of the source code for each exercise, along with a sample output. Leave the source code for all exercises in your directory **csc2405/mainargs**.