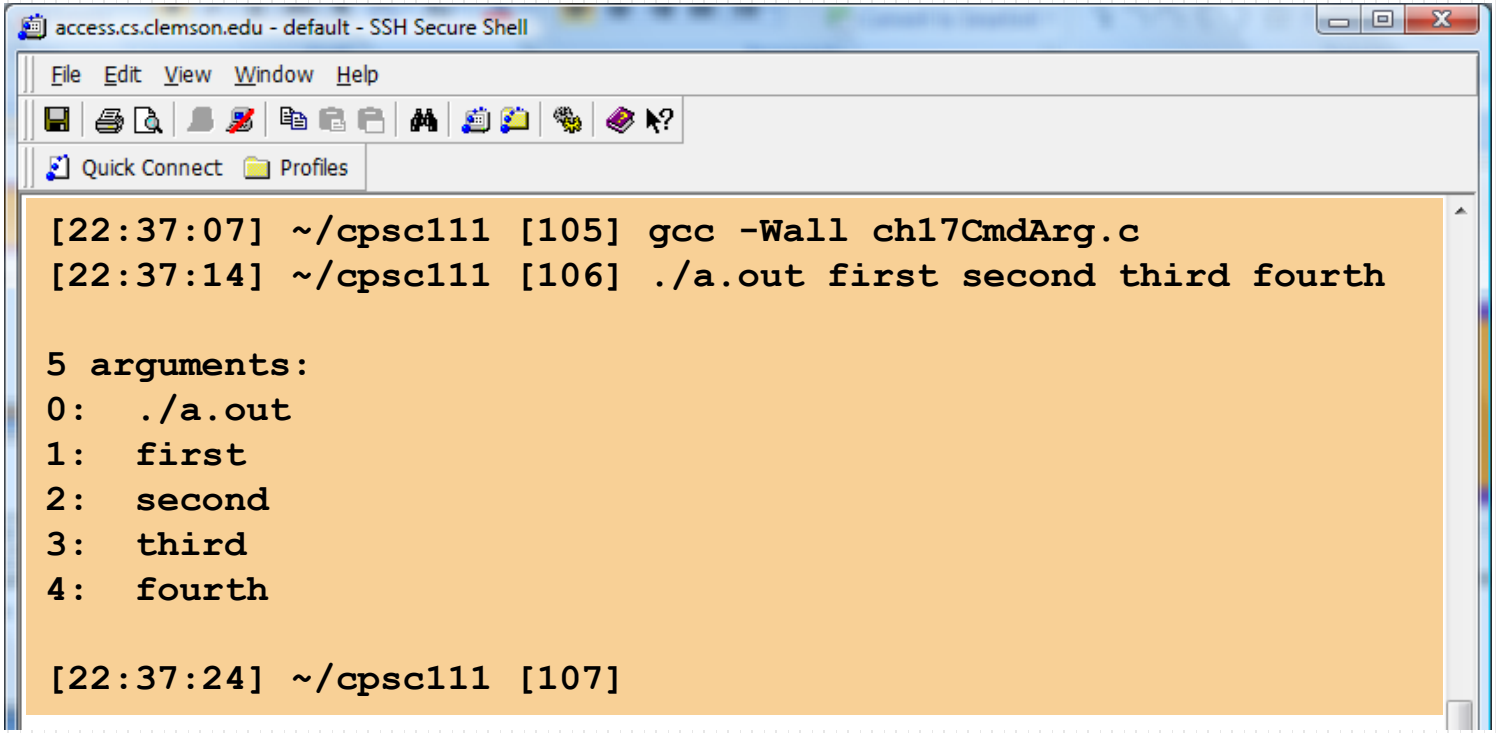
Programming in C

Chapter 16b
Command Line Arguments



Redirection

- Redirection: Read / Write to actual file
  - stdin: **`cmd < input-file`**
    - Ex: **`./a.out < nums.txt`**
  - stdout: **`cmd > output-file`**
    - Ex: **`./a.out > report.txt`**
  - stdout (append): **`cmd >> output-file`**
    - Ex: **`./a.out >> report.txt`**
  - Both: **`cmd < input-file > output-file`**
    - Ex: **`./a.out < nums.txt > report.txt`**

Controlling the Command Line

- Command line arguments
  - It is often useful to pass arguments to a program via the command line. e.g.

```
gcc –Wall myProg.c –lm
```

  - In this case, there are four command line arguments.
    - The count includes the command to execute the program.

Command Line Arguments

- When a program is started from the command line,
  - The character strings (separated by spaces) comprising the program name and the remaining arguments are copied by the operating system into memory space occupied by the new program.
  - A table or array of addresses is passed to the main function. These values can be accessed by the main( ) function via arguments to main():

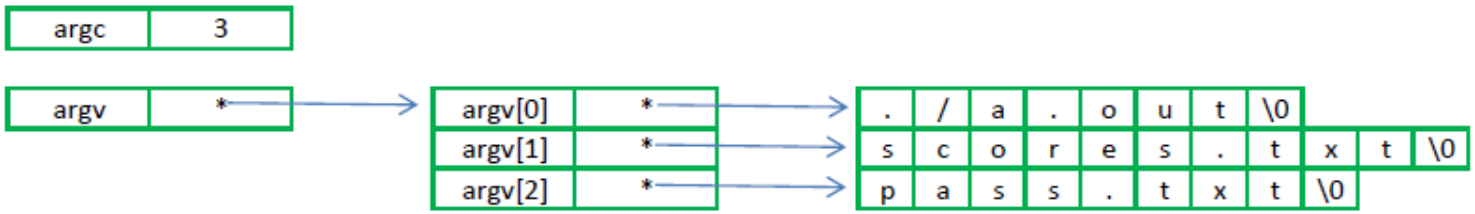
```
int main(int argc,      // count of number of command line arguments
         char *argv[])  // array of pointers to argument strings
```

  - Parameter names argc and argv are used by convention but not required by C.

Command Line Example

- `./a.out scores.txt pass.txt`

```
int main(int argc,      // count of number of command line arguments
         char *argv[])  // array of pointers to argument strings
```



Printing Command Line Arguments

- Example:

```
int main(int argc, char *argv[])
{
    int index = 0;
    while (index < argc) {
        printf("%s\n", argv[index]);
        index++;
    }
}
```

Printing Program Output

- When the program is invoked as follows:
  `./a.out input hello 5 mydata.dat`

- Output is:

```
./a.out
input
hello
5
mydata.dat
```

Add Command Line Numbers

```
int main(int argc, char *argv[])  {
   // variables
   int arg;
   float sum = 0;

   // verify command line arguments
   if (argc < 2) {
      printf("\nNumbers not specified on command line!\n\n");
      exit(1);
   }

   // add numbers
   for (arg = 1; arg < argc; arg++)
      sum += atof(argv[arg]);

   // print sum
   printf("\nSum is %f\n\n", sum);

   return 0;
}
```

atof = alpha (string) to float

```
./a.out 3.5 -1.1 4.725

Sum is 7.125000
```

Programming in C

Chapter 16b
Command Line Arguments

THE END