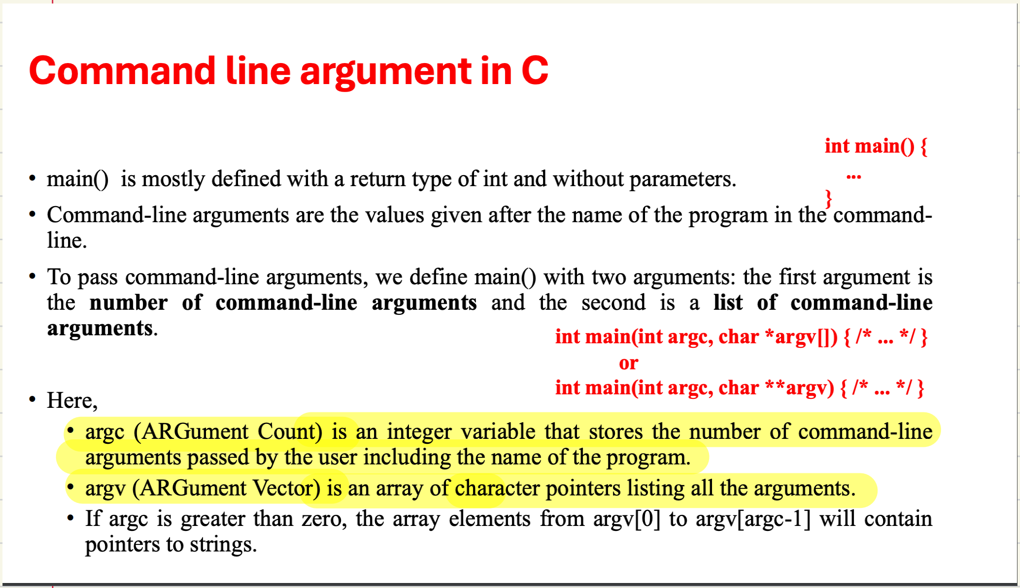


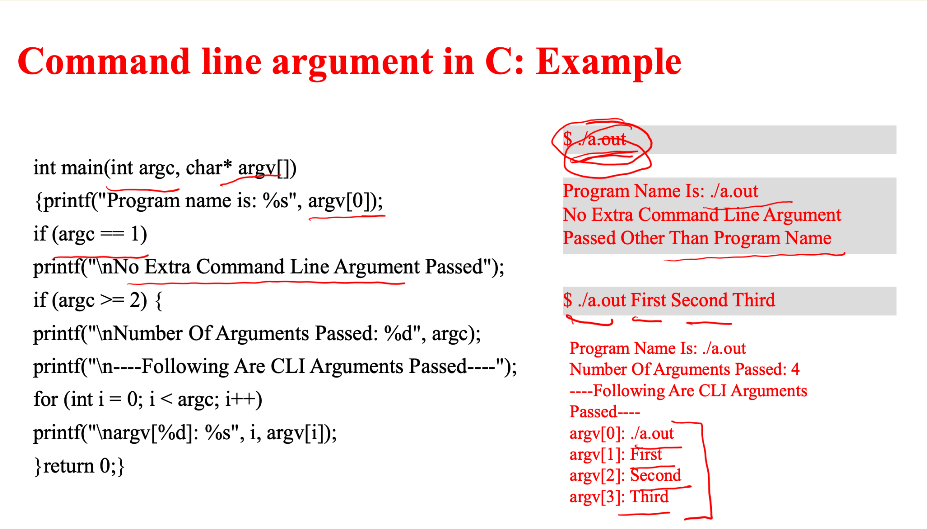
Command line argument in C

```
int main() {
   ...
}
```

- main() is mostly defined with a return type of int and without parameters.
- Command-line arguments are the values given after the name of the program in the command-line.
- To pass command-line arguments, we define main() with two arguments: the first argument is the **number of command-line arguments** and the second is a **list of command-line arguments**.

```
int main(int argc, char *argv[]) { /* ... */ }
or
int main(int argc, char **argv) { /* ... */ }
```

- Here,
  - argc (ARGument Count) is an integer variable that stores the number of command-line arguments passed by the user including the name of the program.
  - argv (ARGument Vector) is an array of character pointers listing all the arguments.
  - If argc is greater than zero, the array elements from argv[0] to argv[argc-1] will contain pointers to strings.



Command line argument in C: Example

```
int main(int argc, char* argv[])
{printf("Program name is: %s", argv[0]);
if (argc == 1)
printf("\nNo Extra Command Line Argument Passed");
if (argc >= 2) {
printf("\nNumber Of Arguments Passed: %d", argc);
printf("\n----Following Are CLI Arguments Passed----");
for (int i = 0; i < argc; i++)
printf("\nargv[%d]: %s", i, argv[i]);
}return 0;}
```

```
$ ./a.out
```

```
Program Name Is: ./a.out
No Extra Command Line Argument
Passed Other Than Program Name
```

```
$ ./a.out First Second Third
```

```
Program Name Is: ./a.out
Number Of Arguments Passed: 4
----Following Are CLI Arguments
Passed----
argv[0]: ./a.out
argv[1]: First
argv[2]: Second
argv[3]: Third
```