C Command Line Arguments (Questions)

Sayak Haldar

IIEST, Shibpur

1. What does argv and argc indicate in command-line arguments?

(Assuming: int main(int argc, char *argv[]))
a) argument count, argument variable
b) argument count, argument vector
c) argument control, argument variable
d) argument control, argument vector

2. Which of the following syntax is correct for command-line arguments?

a) int main(int var, char *varg[])
b) int main(char *argv[], int argc)
c) int main()
```
  {
        int argv, char *argc[];
   }
```
d) Both (a) and (b)

3. In linux, argv[0] by command-line argument can be occupied by

a) ./a.out
b) ./test
c) ./fun.out.out
d) All of the mentioned

4. What type of array is generally generated in Command-line argument?

a) Single dimension array
b) 2-Dimensional Square Array
c) Jagged Array
d) 2-Dimensional Rectangular Array

5. What would be the output if we try to execute following segment of code (assuming the following input “cool brother in city”)?

printf(“%s\n”, argv[argc]);
a) (null)
b) City
c) In
d) Segmentation Fault

6. The first argument in command line arguments is (we are talking about argc)

a) The number of command-line arguments the program was invoked with
b) A pointer to an array of character strings that contain the arguments

c) Nothing
d) Both a & b

7. The second (argument vector) in command line arguments is

a) The number of command-line arguments the program was invoked with
b) A pointer to an array of character strings that contain the arguments, one per string
c) Nothing
d) Both a & b

8. argv[0] in command line arguments is

a) The name by which the program was invoked
b) The name of the files which are passed to the program
c) Count of the arguments in argv[] vector
d) Both a & b

9. A program that has no command line arguments will have argc

a) Zero
b) Negative
c) One
d) Two

10. The index of the last argument in command line arguments is

a) argc – 2
b) argc + 1
c) argc
d) argc – 1

11. What is the output of this C code (if run with no options or arguments)?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      printf("%d\n", argc);
```

```
5.      return 0;
6.  }
```

a) 0
b) 1
c) Depends on the platform
d) Depends on the compiler

12. What is the output of this C code (run without any commandline arguments)?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      while (argc--)
5.      printf("%s\n", argv[argc]);
6.      return 0;
7.  }
```

a) Compile time error
b) Executablefilename
c) Segmentation fault
d) Undefined

13. What is the output of this C code (run without any commandline arguments)?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      printf("%s\n", argv[argc]);
5.      return 0;
6.  }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

14. What is the output of this C code (run without any commandline arguments)?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      while (*argv++ != NULL)
5.      printf("%s\n", *argv);
6.      return 0;
7.  }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

15. What would be the output of the following c code?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      while (*argv != NULL)
5.      printf("%s\n",  *(argv++));
6.      return 0;
7.  }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

16. What is the output of this C code(run without any command line arguments)?

```
1.  #include <stdio.h>
2.  int main(int argc, char *argv[])
3.  {
4.      while (argv != NULL)
5.      printf("%s\n",  *(argv++));
6.      return 0;
7.  }
```

a) Segmentation fault/code crash
b) Executable file name
c) Depends on the platform
d) Depends on the compiler

References:

1. http://www.sanfoundry.com/c-interview-questions-answers/