# Programming Assignment #1

## Your Own Unix Utilities – wzip, wunzip

2024. 09. 25

# Overview

- **Due date: 10/4**
  - No late submission is allowed

- **Objectives**
  - To gain experiences editing, compiling, testing, and debugging C programs under Unix
  - To familiarize yourself with a shell / terminal / command-line of UNIX
  - To Learn a little about how UNIX utilities are implemented

- **Implementation**
  - Build simple versions of UNIX utilities (Your own ***zip*** & ***unzip*** command)
  - We'll call each of them a slightly different name to avoid confusion
    - For example, ***wzip*** & ***wunzip***

# Overview – Your Task

- **Download script for testing the code**
  - AjouBb, download "`HW1_unix-utilities.tar`" file, upload it to server, and then extract it
  - For example, scp -P2222 HW1_unix-utilities.tar <myid>@lagavulin.aisa-lab.ajou.ac.kr:~/
- **Write "wzip.c" and "wunzip.c"**

- **Each should compile successfully when compiled with the -Wall and -Werror**

- **Each should pass the tests**

  If you pass test, can see this command line output

```
prompt> ./test-wzip.sh
test 1: passed
test 2: passed
test 3: passed
test 4: passed
test 5: passed
test 6: passed
```

# Programming Environments

- **Background**
  - List of commands that you might need. Please refer to the man page for the details
    - `fopen()`
    - `fgets()`
    - `fwrite()`
    - `fread()`
    - `fclose()`
    - …
  - `e.g., $> man fopen`

  - Please refer the required header file also

# Programming Environments

- **You will build come in a pair**
  - The first program, **wzip** is a file compression tool
  - The other program, **wunzip**, is a file decompression tool

- **The compression algorithm is called run-length encoding (RLE)**
  - When you encounter n characters of the same alphabet / whatever chars (e.g., aaaaaa) the compression tool (wzip) will turn that into the number n (e.g., 6) and a single instance of the character (e.g., a)
  - So, the final compression result is 6a (4 bytes + 1 byte = 5 bytes)

- **Another example**
  - If we had a file with the following contents: aaaaaaaaaabbbb,
  - the tool would turn it (logically) into: 10a4b

# C Programming Background

- **What is argc & argv in C programming?**
  - ***Argument Count (argc)***: integer variable storing "the number of command-line arguments"
  - ***Argument Vector (argv)***: character array containing "the actual arguments"
- **Note that argv[0] is the name of the program**
- **After that argv[0 … agrc-1] contains the actual command-line arguments**

```c
// C program named mainreturn.c to demonstrate the working
// of command line arguement
#include <stdio.h>

// defining main with arguments
int main(int argc, char* argv[])
{
        printf("You have entered %d arguments:\n", argc);

        for (int i = 0; i < argc; i++) {
                printf("%s\n", argv[i]);
        }
        return 0;
}
```

```
song@eong:~/ex$ vim argc.cpp
song@eong:~/ex$ g++ argc.cpp -o main
song@eong:~/ex$  ./main ajou daxue
You have entered 3 arguments:
./main
ajou
daxue
```

# Program wzip and wunzip

- **CAUTION: The exact compression format is important**
  - <# of appearance: int = 4bytes><single character: char = 1 byte>…
  - Each 5 bytes constructs a single block describing the repeated char stream

- When you program **wzip**,
  - To write out an integer in binary format (not ASCII), you should use **fwrite()**
  - Read the man page for the details

- For **wunzip**,
  - To read the compressed file, you might use **fread()**,
  - Then, print out uncompressed output to STDOUT - **printf()**

- **Also, man & README.md is your friend**

# Usage of wzip and wunzip

- **wzip (Compresion)**

```
prompt> ./wzip file.txt > file.z
```

  Note that “>” sign means redirection to STDOUT (=“1>”)

- **wunzip (Decompression)**

```
prompt> ./wunzip file.z
```

# Program wzip and wunzip

- **Requirements for wzip & wunzip**
  1. Correct invocation should pass one or more files via the command line to the program.
  2. If no files are specified, the program should exit with return code 1 and print the message
     - "wzip: file1 [file2 ...]" (followed by a newline) or "wunzip: file1 [file2 ...]" (followed by a newline) for wzip and wunzip respectively.
  3. If the program encounters a file that it cannot open, the program should exit with return code 1 print the message
     - “wzip: cannot open file”(followed by a newline) or “wunzip: cannot open file”(followed by a newline) for wzip and wunzip
  4. The format of the compressed file must match the description (4-byte integer + 1-byte character)
  5. If multiple files are passed to **wzip**, they are compressed into a single compressed output, and when unzipped, will turn into a single uncompressed stream of text (thus, the information that multiple files were originally input into **wzip** is lost). The same thing holds for **wunzip**.

# Conclusion

- **Enjoy Your First OS Programming!** 
- **Within Due: 10/4**
  - No late submission is allowed