

C Programming II

2023 Spring

Homework 04

Instructor: Po-Wen Chi

Due: 2023.05.30 PM 11:59

Policies:

- **Zero tolerance** for late submission.
- **Plagiarism is not allowed.** Both source and copycat will be **zero**.
- You need to prepare a README file about how to make and run your program. Moreover, you need to provide your name and your student ID in the README file.
 - Your Name and Your ID.
 - The functional description for each code.
 - Anything special.
- Please pack all your submissions in one zip file.
- For convenience, your executable programs must be named following the rule hw**XXYY**, where the red part is the homework number and the blue part is the problem number. For example, **hw0102** is the executable program for homework #1 problem 2.
- I only accept **PDF**. MS Word is not allowed.
- **Do not forget your Makefile.** For convenience, each assignment needs only one Makefile.

1 Process Monitor (20 pts)

The proc filesystem (**procfs**) is a special filesystem in Unix-like operating systems that presents information about processes and other system information. It provides a more convenient and standardized method for accessing process data **held in the kernel** than traditional tracing methods or direct access to kernel memory. How to access a process information? First, you have to have **pid** of the given process. Then go to **/proc/[pid]**. The following is an example for a process where pid is 1. Here **stat** is the process information. You can see that you can access the information as processing a text file.

```

1 $ ls /proc/1
2 arch_status  clear_refs      cwd      gid_map  maps      net
   oom_score_adj  root      smaps      status      uid_map
3 attr          cmdline      environ  io          mem        ns
   pagemap        sched      smaps_rollup  syscall      wchan
4 autogroup      comm        exe      limits      mountinfo  numa_maps
   patch_state    schedstat  stack      task
5 auxv          coredump_filter  fd        loginuid  mounts      oom_adj
   personality    sessionid  stat        timers
6 cgroup        cpuset      fdinfo     map_files  mountstats  oom_score
   projid_map     setgroups  statm        timerslack_ns
7 $ cat /proc/1/stat
8 1 (systemd) S 0 1 1 0 -1 4194560 318496 22492543 105 8615 835 678 342622
   126871 20 0 1 0 10 174039040 3054 18446744073709551615 1 1 0 0 0 0
   671173123 4096 1260 0 0 0 17 7 0 0 405 0 0 0 0 0 0 0 0

```

Wait!! I cannot interpret these ugly numbers. Do not worry. You can read manual from **man 5 proc**. Now I want you to develop a process monitor as follows.

```

1 $ ./hw0401 --help
2 Usage: hw0401 [options]... <pid>
3 Options:
4   -i, --interval <int>      Display the process information every <int>
   seconds.
5                               Default value: 1, Range: 1-100
6   -c, --count <count>       Display the process information <count> times.
7                               Default behavior: infinite loop, Range: 1-1000
8   -t, --timestamp            Display the timestamp.
9   -h, --help                  Display this description.
10 $ ./hw0401 -c 1 7761
11 7761: 1.0(%CPU) 4.1(%MEM) chrome(COMMAND)
12 $ ./hw0401 -t
13 2023-05-04 02:34:56 7761: 1.0(%CPU) 4.1(%MEM) chrome(COMMAND)
14 2023-05-04 02:34:57 7761: 1.0(%CPU) 4.2(%MEM) chrome(COMMAND)
15 ...

```

For your simplicity, I give you some hints:

- Memory usage for a process is its **vsz**.
- CPU usage is the sum of **utime** and **stime**. Note that the unit is not "second" but clock ticks.

2 Useful Macros (20 pts)

Please implement the following macros in **macros.h**. TAs will include macros in **hw0402.c**. **Again, do not forget to build hw0402.c**. Since they are macros, you do not need to worry if the user uses this correctly.

```

1 /* Bits related Macros */
2
3 // Return 2 to x. You do not need to worry the overflow issue.
4 #define BIT(x)

```

```

5 // Set x's p-th bit to 1. Count from 0 and from LSB.
6 #define SETBIT(x,p)
7 // Set x's p-th bit to 0. Count from 0 and from LSB.
8 #define CLEARBIT(x,p)
9 // Get x's p-th bit to 1. Count from 0 and from LSB.
10 #define GETBIT(x,p)
11 // Toggle x's p-th bit to 1. Count from 0 and from LSB.
12 #define TOGGLEBIT(x,p)
13
14 /* Loop related Macros */
15 // Example: RANGE (i, 10, 20)
16 // it will be 10,11,12,13,14,15,16,17,18,19,20
17 // Example: RANGE (i, 5, -5)
18 // it will be 5,4,3,2,1,0,-1,-2,-3,-4,-5
19 #define RANGE(i,y,x)
20
21 // FOREACH (item, array)
22 // A C implementation of foreach loop, it will go through each element of an
    array, and will perform operations on each element as returned into
    pointer item variable.
23 #define FOREACH(i,A)

```

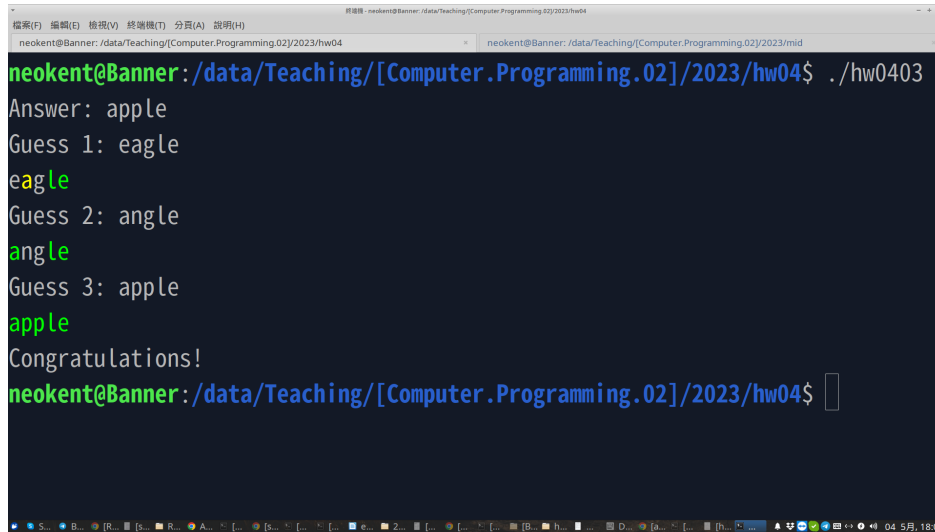
3 Wordle (20 pts)

Have you ever played **wordle**? If not, you can see the following url.

<https://zh.wikipedia.org/zh-tw/Wordle>

This time, I want you to develop this game.

1. Install a package called **wamerican**. This is a dictionary package. The dictionary will be installed on `/usr/share/dict/`. I guarantee that TAs' computers will install this package.
2. Open `/usr/share/dict/american-english` and find all **five-letter words**. Note that you should **ignore all capitalize words and words with suffix**.
3. Randomly pick one five-letter word and **print the word**. This is for TA to justify your program.
4. Give the user six chances to guess the word. **The guess must also be in the dictionary**. After every guess, each letter is marked as either green, yellow or white. For your simplicity, I promise all inputs are lower case and are definitely five-letter words.
 - green indicates that letter is correct and in the correct position.
 - yellow means it is in the answer but not in the right position.
 - white indicates it is not in the answer at all.
 - Multiple instances of the same letter in a guess, such as the "o"s in "robot", will be colored green or yellow only if the letter also appears multiple times in the answer; otherwise, excess repeating letters will be colored gray.



```
neokent@Banner: /data/Teaching/[Computer.Programming.02]/2023/hw04$ ./hw0403
Answer: apple
Guess 1: eagle
eagle
Guess 2: angle
angle
Guess 3: apple
apple
Congratulations!
neokent@Banner: /data/Teaching/[Computer.Programming.02]/2023/hw04$
```

Figure 1: Wordle example.

5. If the user wins the game, print "Congratulations"; otherwise, print "Good Luck".

Figure 1 is an example.

4 Data Hiding (20 pts)

You all know that there is a **file size** field in the BMP header, right? What will happen if the real file size is larger than the value in the file size field? You can do an experiment with the following command.

```
1 $ cat maldives.bmp exam_solution.pdf > output.bmp
```

You will find that **output.bmp** can be opened by the image viewer and we hide a pdf file in the image. This time, I want you to implement the same feature. First, the user can append many files in a BMP file. The file structure is shown in Fig. 2.

The operation is as follows.

```
1 # Hide test.pdf to maldives.bmp
2 $ ./hw0404 -i test.pdf maldives.bmp
3 # Extract all hidden files in maldives.bmp
4 $ ./hw0404 -e maldives.bmp
```

5 Are you a leek? (20 pts)

Smallten has recently started to dabble in the stock market and, unsurprisingly, has become a typical "leek" due in large part to his tendency to buy and sell without looking at prices. Therefore, in order to rescue Smallten from his impending financial ruin, he hopes that you can help him develop a program to query historical prices and determine what a reasonable price would be.

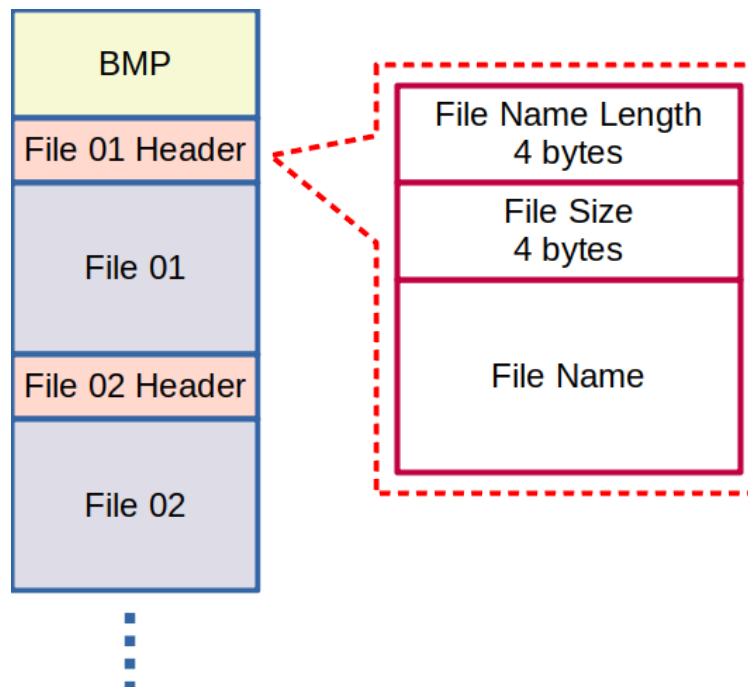


Figure 2: Data Hiding in BMP.

Please use the following website and libraries to achieve the goal:

- Finance information: <https://finance.yahoo.com>
- libcurl: <https://curl.se/libcurl/>
- libxml2: <https://github.com/GNOME/libxml2>
- cJSON: <https://github.com/DaveGamble/cJSON>

TAs will install libcurl and libxml2 on their computer. As for cJSON, since there is only one .c and .h file, please add it to your folder yourself.

The program usage should be

```
1 $ ./hw0405 --help
2     -s : symbol of stock
3     -i : input JSON file name
4     -o : output JSON file name
5 $ ./hw0405 -s 2357.TW -i date.json -o price.json
```

Example of Input and Output file

- date.json

```

1  [
2    {
3      "start": "2023-04-01",
4      "end": "2023-04-20"
5    },
6    {
7      "start": "2022-10-02",
8      "end": "2022-10-02"
9    }
10   ...
11 ]

```

- price.json

```

1  [
2    {
3      "data": [
4        {
5          "date": "2023-04-06",
6          "Open": 269.50,
7          "High": 270.50,
8          "Low": 268.50,
9          "Close": 270.50,
10         "Volume": 1635045
11        },
12        ...
13        {
14          "date": "2023-04-19",
15          "Open": 279.00,
16          "High": 281.00,
17          "Low": 278.00,
18          "Close": 278.50,
19          "Volume": 1535742
20        }
21      ],
22      "max": 282.00,
23      "min": 268.50
24    },
25    {
26      "date": null,
27      "max": null,
28      "min": null
29    }
30   ...
31 ]

```

- "max" is the maximum value of the "High" among all the data.
- "min" is the minimum value of the "Low" among all the data.
- If no data is found within the interval, its value is null.

6 Bonus: Macro (5 pts)

Please read the following code.

```
1 #include <stdio.h>
2
3 #define Peval(cmd) printf( #cmd ": %g\n", cmd );
4
5 int main()
6 {
7     double *plist = (double[]){1, 2, 3};
8     double list[] = {1, 2, 3};
9     Peval( sizeof( plist ) / ( sizeof( double ) + 0.0 ) );
10    Peval( sizeof( list ) / ( sizeof( double ) + 0.0 ) );
11    return 0;
12 }
```

Please run this code and **explain this macro**.

7 Bonus: File (5 pts)

In Linux, there is a command called **file**. This command can be used to check the file type. There are some examples.

```
1 $ file 123.txt
2 123.txt: ASCII text, with no line terminators
3 $ file jise_foreword.docx
4 jise_foreword.docx: Microsoft Word 2007+
5 $ file C302.pdf
6 C302.pdf: PDF document, version 1.5
7 $ file a.out
8 a.out: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically
   linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=
   e8d878ec62bfb1fd52053e889eeec46304f17d2, for GNU/Linux 3.2.0, not
   stripped
```

Could you please explain how this command can determine the file type?