

HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

Faculty of Computer Science & Engineering



Operating System

Lab 1

Instructor: Mrs. Le Thanh Van

Students: Cao Hoang Kiet - 2053165

Ho Chi Minh City, February 2022



Contents

1 Questions	2
1.1 What are the advantages of Makefile? Give examples?	2
1.1.1 Advantages	2
1.1.2 Examples	2
1.2 In case of source code files located in different places, how can we write a Makefile?	2
1.3 What the output will be at LINE A? Explain your answer.	3
2 Basic Commands	4
2.1 Create a new directory entitled with your student ID + Create a new file example.txt under folder <StudentID>	4
2.2 List the content of <student-ID> directory + View the content of file example.txt + Show first 5 lines from example.txt + Show last 5 lines from example.txt.	5
3 Programming	6
3.1 Problem 1	6
3.2 Problem 2	6
3.3 Problem 3	7
3.4 Problem 4	7



1 Questions

1.1 What are the advantages of Makefile? Give examples?

1.1.1 Advantages

- It makes codes more concise and clear to read and debug.
- No need to compile entire program every time whenever you make a change to a functionality or a class. Makefile will automatically compile only those files where change has occurred.
- Generally, in long codes or projects, Makefile is widely used in order to present project in more systematic and efficient way.

1.1.2 Examples

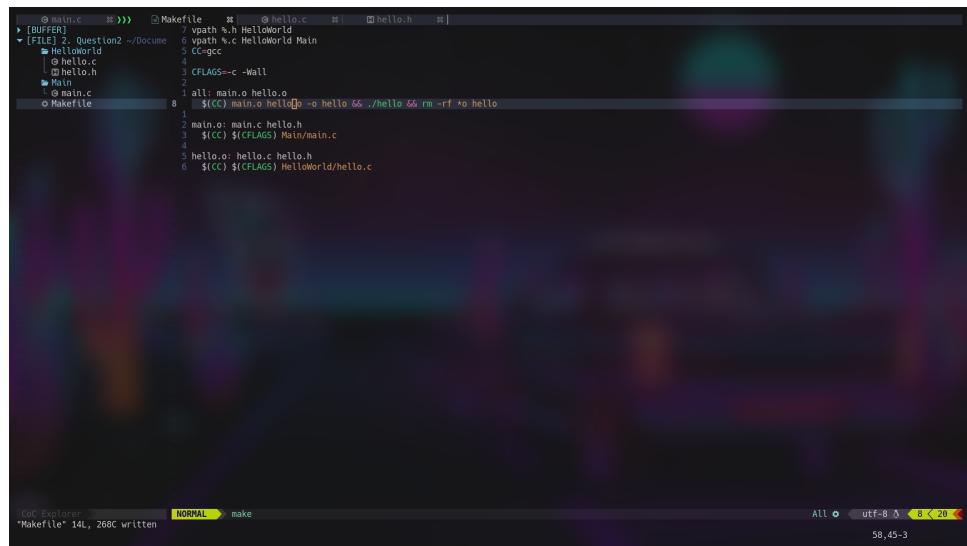
```
CC=gcc
CFLAGS=-c -Wall
all: main.o hello.o
$(CC) main.o hello.o -o hello
$(CFLAGS) main.c
$(CC) $(CFLAGS) hello.c
$(CC) $(CFLAGS) hello.c

void helloworld(void);

main.o: main.c hello.h
$(CC) $(CFLAGS) main.c
hello.o: hello.c hello.h
$(CC) $(CFLAGS) hello.c
```

1.2 In case of source code files located in different places, how can we write a Makefile?

We use vpath to specify where files exist

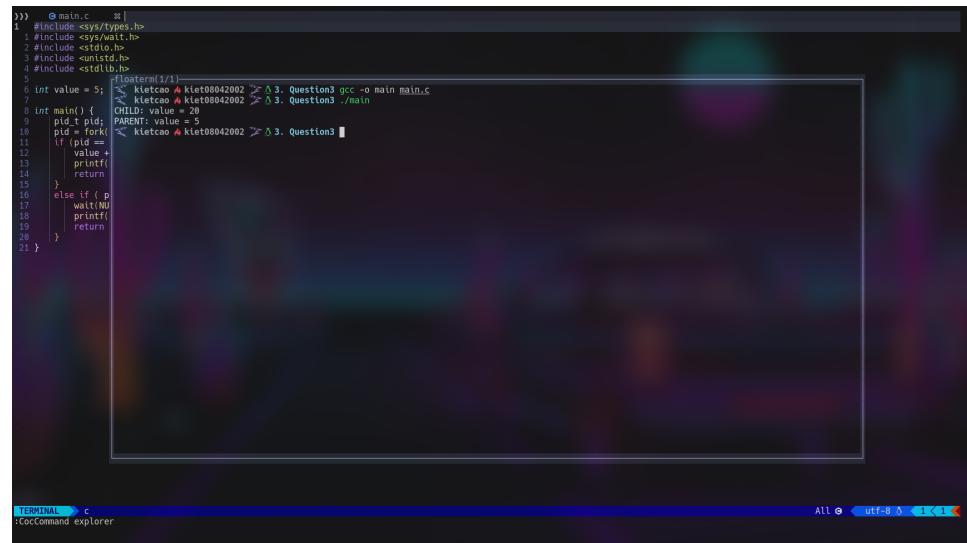


```
① Hello.c ② Makefile ③ hello.c ④ hello.h ⑤
⑥ [FILE] ⑦ Question2 ~/Documents ⑧ vpath %.c HelloWorld Main
⑨ HelloWorld
⑩ @ hello.c
⑪ @ hello.h
⑫ @ Main
⑬ @ main.c
⑭ @ Makefile
⑮
⑯ ① all: main.o hello.o
⑰ ② $(CC) main.o hello.o -o hello && rm -rf *o hello
⑱ ③ $(CC) $(CFLAGS) Main/main.c
⑲ ④
⑳ ⑤ hello.o: hello.c Hello.h
⑳ ⑥ $(CC) $(CFLAGS) HelloWorld/hello.c
⑷
⑵
⑴
```

Code Explorer NORMAL make

All • utf-8 △ 8 < 20 58,45-3

1.3 What the output will be at LINE A? Explain your answer.



```
① #include <sys/types.h>
② #include <sys/wait.h>
③ #include <sys/types.h>
④ #include <sys/stat.h>
⑤ #include <unistd.h>
⑥ #include <stdlib.h>
⑦
⑧ int value = 5;
⑨
⑩ void f(int value) {
⑪     if (value > 3) {
⑫         value = 5;
⑬     }
⑭     printf("value = %d\n", value);
⑮ }
⑯
⑰ int main() {
⑱     pid_t pid;
⑲     pid = fork();
⑳     if (pid < 0) {
⑳         value + 5;
⑳         printf("value = %d\n", value);
⑳         return 1;
⑳     }
⑳     else if (pid == 0) {
⑳         value = 5;
⑳         wait(NULL);
⑳         printf("value = %d\n", value);
⑳         return 0;
⑳     }
⑳ }
⑳ }
```

CodeCommander C CodeCommander explorer

All • utf-8 △ 1 < 1

Answer: The output at LINE A is still 5 because the child just updates its copy of value. So when control returns to the parent, its value is keep at 5.



2 Basic Commands

- 2.1 Create a new directory entitled with your student ID + Create a new file example.txt under folder <StudentID>

```
kietcao ~ kiet08042002 ~/lab1 ls
└── Basic/   └── Programming/   └── Questions/
kietcao ~ kiet08042002 ~/lab1 cd Basic
kietcao ~ kiet08042002 ~/Basic ls
Nothing to show here
kietcao ~ kiet08042002 ~/Basic mkdir 2053165
kietcao ~ kiet08042002 ~/Basic ls
└── 2053165/
kietcao ~ kiet08042002 ~/Basic cd 2053165
kietcao ~ kiet08042002 ~/2053165 ls
Nothing to show here
kietcao ~ kiet08042002 ~/2053165 nvim example.txt
kietcao ~ kiet08042002 ~/2053165 ls
└── example.txt
```



- 2.2 List the content of <student-ID> directory + View the content of file example.txt + Show first 5 lines from example.txt + Show last 5 lines from example.txt.

```
⚡ kietcao 🔥 kiet08042002 ⚡ 2053165 ls
  └── example.txt
⚡ kietcao 🔥 kiet08042002 ⚡ 2053165 cat example.txt
Name: Cao Hoang Kiet
Student Id: 2053165
Age: 20
Year: 2022
School: HCMUT
Class OS: CC02
This is Basic For Lab 1
⚡ kietcao 🔥 kiet08042002 ⚡ 2053165 head -5 example.txt
Name: Cao Hoang Kiet
Student Id: 2053165
Age: 20
Year: 2022
School: HCMUT
⚡ kietcao 🔥 kiet08042002 ⚡ 2053165 tail -5 example.txt
Age: 20
Year: 2022
School: HCMUT
Class OS: CC02
This is Basic For Lab 1
⚡ kietcao 🔥 kiet08042002 ⚡ 2053165
```

3 Programming

3.1 Problem 1

```
))> factorial.c  => factorial.h => [FILE] Problem 1+2+3 ~/Docum
-> Factorial
  factorial.c
  factorial.h
  factorial.h
  main.c
  Makefile

1 #ifndef FATORIAL_H
2 #define FATORIAL_H
3 int factorial (const int nNumber);
4
5 #endif
```

3.2 Problem 2

3.3 Problem 3

```
↳ factorial.c  ||  ↳ readline.c  ||  ↳ main.c  ||  ↳ factorial.h  ||  ↳ [BUPFILE] Problem 1+2+3 ~/Docum
↳ [FILE] Problem 1+2+3 ~/Docum
↳ ↳ factorial
↳ ↳ factorial.c
↳ ↳ factorial.h
↳ ↳ readline
↳ ↳ readline.c
↳ ↳ readline.h
↳ ↳ main.c
↳ ↳ Makefile
1 #include <stdio.h>
2 #include <string.h>
3 #include "readline/readline.h"
4 #include "factorial/factorial.h"
5 #define MAX_LIMIT 58
6
7 int main(int argc, char* argv[]) {
8     char str[MAX_LIMIT];
9     printf("Enter a number: ");
10    while (scanf("%s", str)) {
11        if (strlen(str) <= MAX_LIMIT) {
12            if (read_line(str)) {
13                printf("Factorial(%s): %d\n", str, factorial(atol(str)));
14            }
15        }
16        else {
17            printf("-1\n");
18        }
19    }
20    else {
21        printf("-1\n");
22    }
23 }
24
25 return 0;
26 }
```

3.4 Problem 4



```
]]> @ main.c  ==  numbers.txt  01
]]> int main(int argc, char **argv)
]]> {
]]>     char line[MAX_LINE_LENGTH] = {0};
]]>     unsigned int sizeArr = 0;
]]>     int arr[MAX_LINE_LENGTH], num1 = 0;
]]>     FILE *file = fopen("numbers.txt", "r+");
]]>     if (!file)
]]>     {
]]>         perror("numbers.txt");
]]>         return EXIT_FAILURE;
]]>     }
]]>     while (fgets(line, MAX_LINE_LENGTH, file))
]]>     {
]]>         sizeArr++;
]]>         arr[sizeArr-1] = atoi(line);
]]>     }
]]>     if (fclose(file))
]]>     {
]]>         perror("numbers.txt");
]]>         return EXIT_FAILURE;
]]>     }
]]>     pid_t child_pid;
]]>     child_pid = fork();
]]>     if (child_pid == -1)
]]>     {
]]>         perror("Fork failed");
]]>         exit(1);
]]>     }
]]>     if (child_pid == 0) {
]]>         countDivisibleNumbers(arr, sizeArr, &num1, 3);
]]>         num1 != 0 ? printf("%d\n", num1) : -1;
]]>         fflush(stdout);
]]>     }
]]>     else {
]]>         countDivisibleNumbers(arr, sizeArr, &num1, 2);
]]>         num1 != 0 ? printf("%d\n", num1) : -1;
]]>         fflush(stdout);
]]>     }
]]> }
]]> NORMAL  c
]]> Bot 0  utf-8  25 < 5  [21]trailing
]]>
```

```
]]> @ main.c  ==  numbers.txt  01
]]> [FILE] Problem 4  ~/Documents  20 #include <string.h>
]]> [FILE] Problem 4  ~/Documents  20 #include <sys/types.h>
]]> @ main.c  19 #include <sys/wait.h>
]]> @ Makefile  18 #include <unistd.h>
]]> = numbers.txt  17 #define MAX_LINE_LENGTH 50
]]> = floaterm(1/1)
]]> < kietcao  kiet08042002 > Problem 4 make all
]]> gcc -c main.c
]]> gcc main.o -o main && rm -rf main
]]> 2
]]> 3
]]> < kietcao  kiet08042002 > Problem 4
]]>
]]> 21     if (child_pid == -1) {
]]> 22         perror("Fork failed");
]]> 23         exit(1);
]]> 24     }
]]> C/C Explorer  TERMINAL  c
]]> 13 0  utf-8  24 < 5  [21]trailing
]]> 11,42-3
]]>
```