HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

Faculty of Computer Science & Engineering

Operating System
Lab 1

Instructor: Mrs. Le Thanh Van

Students: *Cao Tuan Kiet - 2053166*

Ho Chi Minh City, February 2022

Contents

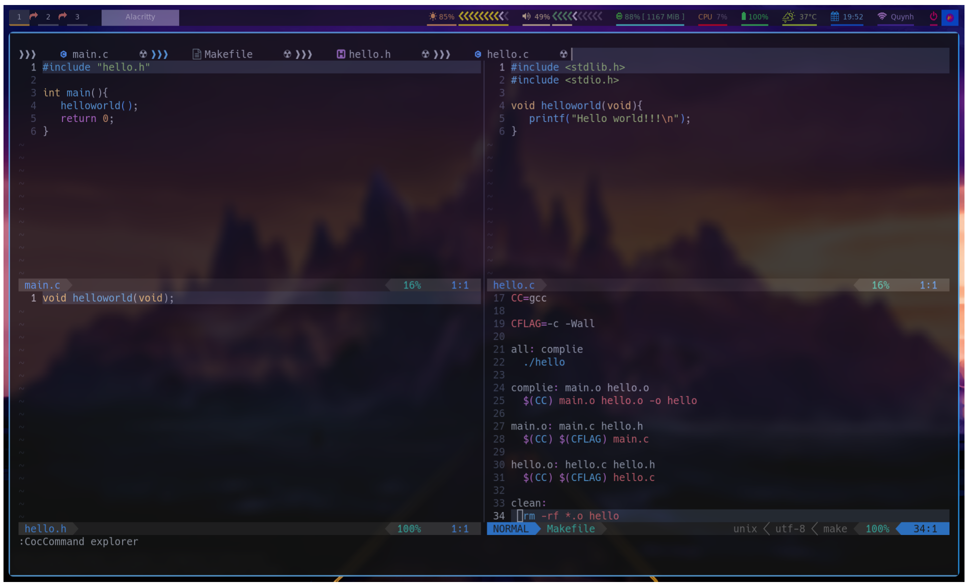
1 Questions

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 Example



main.c
```c
#include "hello.h"

int main(){
   helloworld();
   return 0;
}
```

hello.h
```c
void helloworld(void);
```

hello.c
```c
#include <stdlib.h>
#include <stdio.h>

void helloworld(void){
   printf("Hello world!!!\n");
}
```

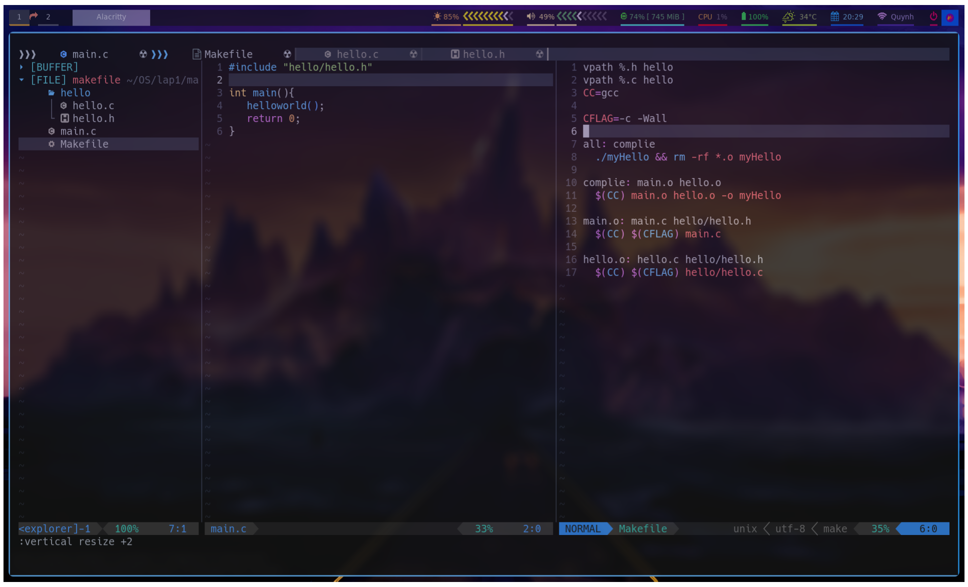
Makefile
```make
CC=gcc

CFLAG=-c -Wall

all: complie
	./hello

complie: main.o hello.o
	$(CC) main.o hello.o -o hello

main.o: main.c hello.h
	$(CC) $(CFLAG) main.c

hello.o: hello.c hello.h
	$(CC) $(CFLAG) hello.c

clean:
	rm -rf *.o hello
```

```
:CocCommand explorer
```

Figure 1: Example of make-file

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

We can use vpath to specify the search path directories for file names. For example:



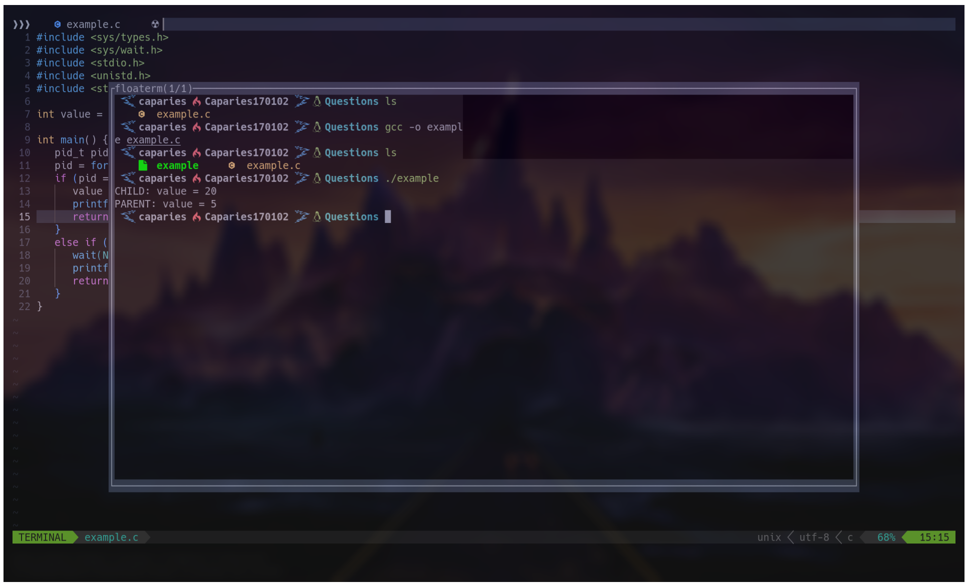
```
#include "hello/hello.h"

int main(){
    helloworld();
    return 0;
}
```

```
vpath %.h hello
vpath %.c hello
CC=gcc

CFLAG=-c -Wall

all: complie
	./myHello && rm -rf *.o myHello

complie: main.o hello.o
	$(CC) main.o hello.o -o myHello

main.o: main.c hello/hello.h
	$(CC) $(CFLAG) main.c

hello.o: hello.c hello/hello.h
	$(CC) $(CFLAG) hello/hello.c
```

```
:vertical resize +2
```

Figure 2: Example of using **vpath** makefile

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



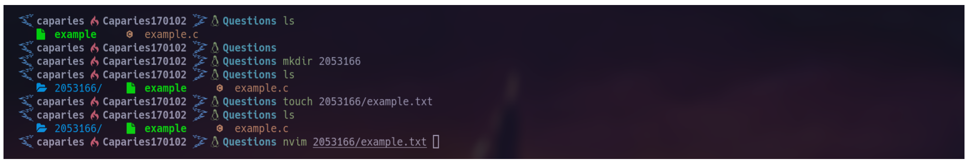
```
#include <sys/types.h>
#include <sys/wait.h>
#include <stdio.h>
#include <unistd.h>
#include <st

int value =

int main() {
    pid_t pid
    pid = for
    if (pid =
        value
        printf
        return
    }
    else if (
        wait(N
        printf
        return
    }
}
```

```
caparies Caparies170102 Questions ls
   example.c
caparies Caparies170102 Questions gcc -o exampl
e example.c
caparies Caparies170102 Questions ls
   example      example.c
caparies Caparies170102 Questions ./example
CHILD: value = 20
PARENT: value = 5
caparies Caparies170102 Questions
```

Figure 3: Example of using **vpath** makefile

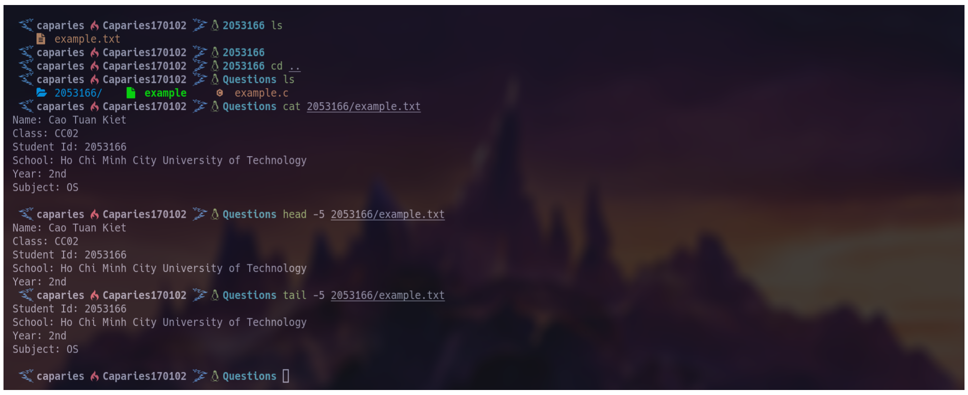
The output is 5 because the value of parent isn't affected by the child process.

2 Basic commands



```
caparies Caparies170102 Questions ls
  example      example.c
caparies Caparies170102 Questions
caparies Caparies170102 Questions mkdir 2053166
caparies Caparies170102 Questions ls
  2053166/     example      example.c
caparies Caparies170102 Questions touch 2053166/example.txt
caparies Caparies170102 Questions ls
  2053166/     example      example.c
caparies Caparies170102 Questions nvim 2053166/example.txt
```

Figure 4: Example of creating new folder, new file and listing contents of a folder

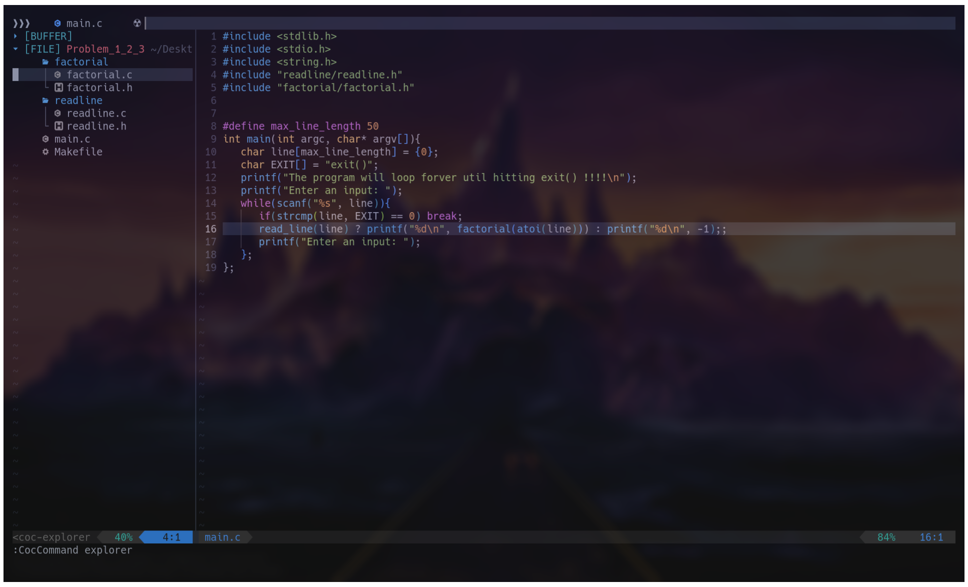


```
caparies Caparies170102 2053166 ls
  example.txt
caparies Caparies170102 2053166
caparies Caparies170102 2053166 cd ..
caparies Caparies170102 Questions ls
  2053166/     example      example.c
caparies Caparies170102 Questions cat 2053166/example.txt
Name: Cao Tuan Kiet
Class: CC02
Student Id: 2053166
School: Ho Chi Minh City University of Technology
Year: 2nd
Subject: OS

caparies Caparies170102 Questions head -5 2053166/example.txt
Name: Cao Tuan Kiet
Class: CC02
Student Id: 2053166
School: Ho Chi Minh City University of Technology
Year: 2nd
caparies Caparies170102 Questions tail -5 2053166/example.txt
Student Id: 2053166
School: Ho Chi Minh City University of Technology
Year: 2nd
Subject: OS

caparies Caparies170102 Questions
```

Figure 5: Example of listing contents of a file

3 Programming

3.1 Problem 1, 2 and 3

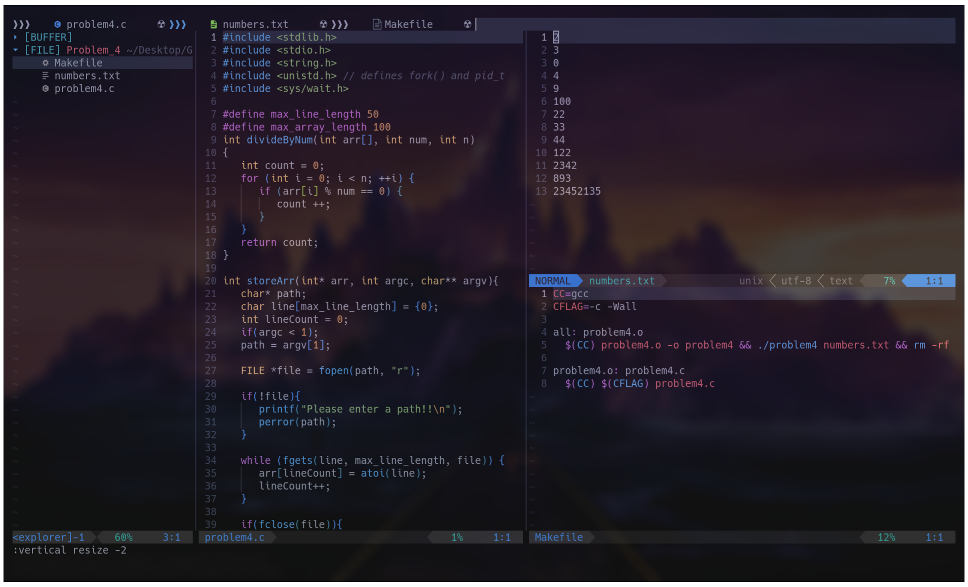


```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "readline/readline.h"
#include "factorial/factorial.h"


#define max_line_length 50
int main(int argc, char* argv[]){
   char line[max_line_length] = {0};
   char EXIT[] = "exit()";
   printf("The program will loop forver util hitting exit() !!!!\n");
   printf("Enter an input: ");
   while(scanf("%s", line)){
      if(strcmp(line, EXIT) == 0) break;
      read_line(line) ? printf("%d\n", factorial(atoi(line))) : printf("%d\n", -1);;
      printf("Enter an input: ");
   };
};
```

```
<coc-explorer  40%  4:1  main.c                                        84%  16:1
:CocCommand explorer
```

Figure 6: Code of problem 1, 2 and 3

3.2 Problem 4



```
:vertical resize -2
```

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h> // defines fork() and pid_t
#include <sys/wait.h>

#define max_line_length 50
#define max_array_length 100
int divideByNum(int arr[], int num, int n)
{
    int count = 0;
    for (int i = 0; i < n; ++i) {
        if (arr[i] % num == 0) {
            count ++;
        }
    }
    return count;
}

int storeArr(int* arr, int argc, char** argv){
    char* path;
    char line[max_line_length] = {0};
    int lineCount = 0;
    if(argc < 1);
    path = argv[1];

    FILE *file = fopen(path, "r");

    if(!file){
        printf("Please enter a path!!\n");
        perror(path);
    }

    while (fgets(line, max_line_length, file)) {
        arr[lineCount] = atoi(line);
        lineCount++;
    }

    if(fclose(file)){
```

```
2
3
0
4
9
100
22
33
44
122
2342
893
23452135
```

```makefile
CC=gcc
CFLAG=-c -Wall

all: problem4.o
	$(CC) problem4.o -o problem4 && ./problem4 numbers.txt && rm -rf

problem4.o: problem4.c
	$(CC) $(CFLAG) problem4.c
```

Figure 7: Code of problem 4