Northeastern University
College of Engineering
Department of Electrical & Computer Engineering

EECE7376: Operating Systems: Interface and Implementation

Homework 1

Problem 1

Approach Summary:

print_args() function iterates over the arguments using a loop and prints each one.

Screenshot:

```
❯ make clean && make
rm -f pr1 pr2 pr3a pr3b pr3c pr4
killall pr1 pr2 pr3a pr3b pr3c pr4 > /dev/null 2>&1
make: [Makefile:15: clean] Error 1 (ignored)
gcc hw1pr1.c -o pr1
gcc hw1pr2.c -o pr2
gcc hw1pr3a.c -o pr3a
gcc hw1pr3b.c -o pr3b
gcc hw1pr3c.c -o pr3c
gcc hw1pr4.c -o pr4
❯ ./pr1 qazwsxe dccrfvv tgvbythby juy
argv[0] = './pr1'
argv[1] = 'qazwsxe'
argv[2] = 'dccrfvv'
argv[3] = 'tgvbythby'
argv[4] = 'juy'
❯ ./pr1
argv[0] = './pr1'
❯ ./pr1 a b c
argv[0] = './pr1'
argv[1] = 'a'
argv[2] = 'b'
argv[3] = 'c'

/work/EECE7376/HW1  main
```

Problem 2

Approach Summary:

Use fgets() to read a string from the user and then print it back, removing the newline character at the end.

Screenshot:

```
❯
❯
❯ ./pr2
$ Hello
Hello
$ How are you  How are you?
How are you  How are you?
$ ^C
❯
❯
/work/EECE7376/HW1  main
```

Problem 3a

Approach Summary:

The child process reads an integer and exits with it as the status, the parent process capture the return value using wait() and then prints.

Screenshot:

```
❯
❯ ./pr3a
Enter a number: 34
Child exited with status 34
❯ ./pr3a
Enter a number: 42
Child exited with status 42
❯ ./pr3a
Enter a number: 233
Child exited with status 233
/work/EECE7376/HW1  main
```

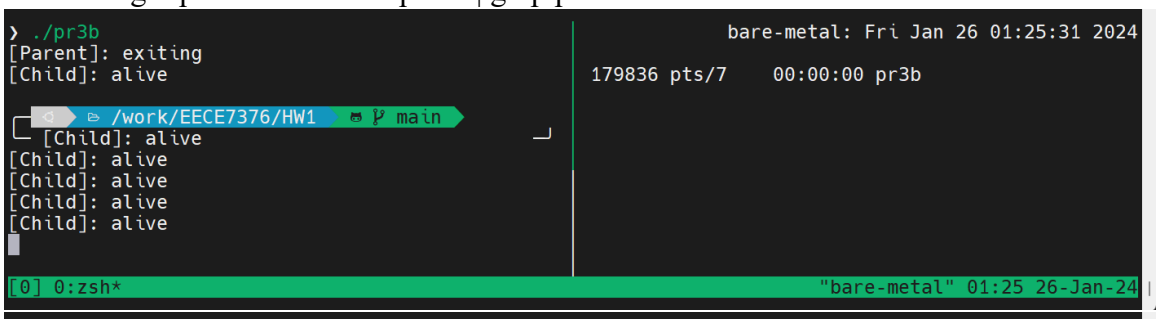
Problem 3b

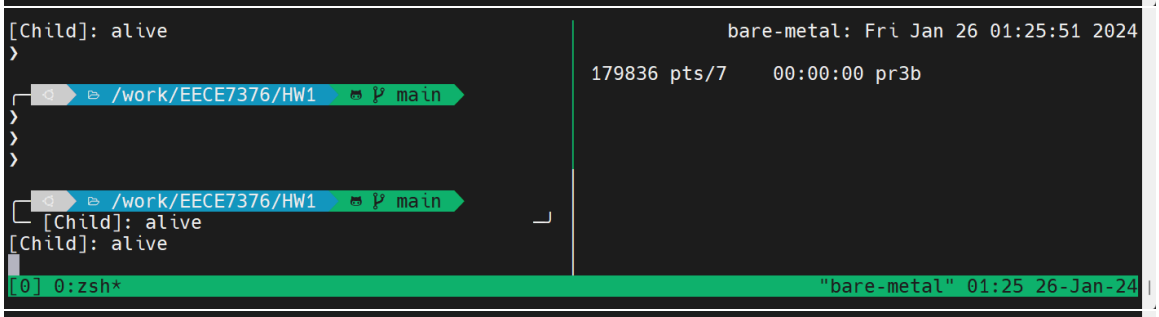
Approach Summary:

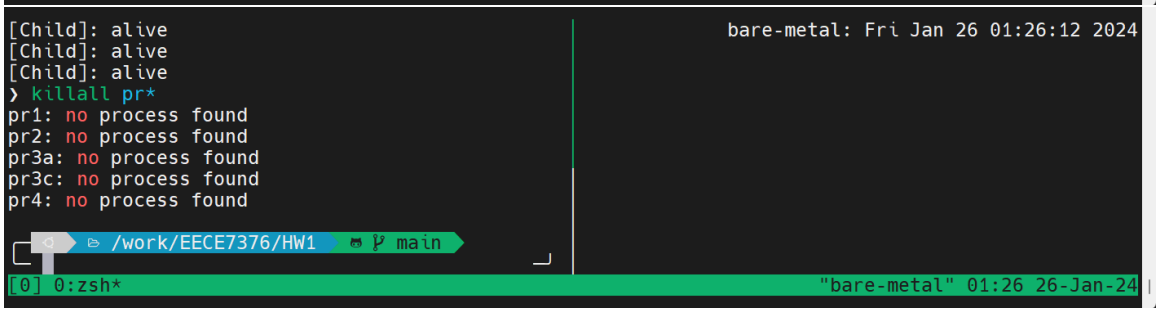
The program creates an orphan process by making the parent exit immediately while the child continues to run, printing a message every 2 seconds.

Screenshot:

Right panel command: ps -A | grep pr







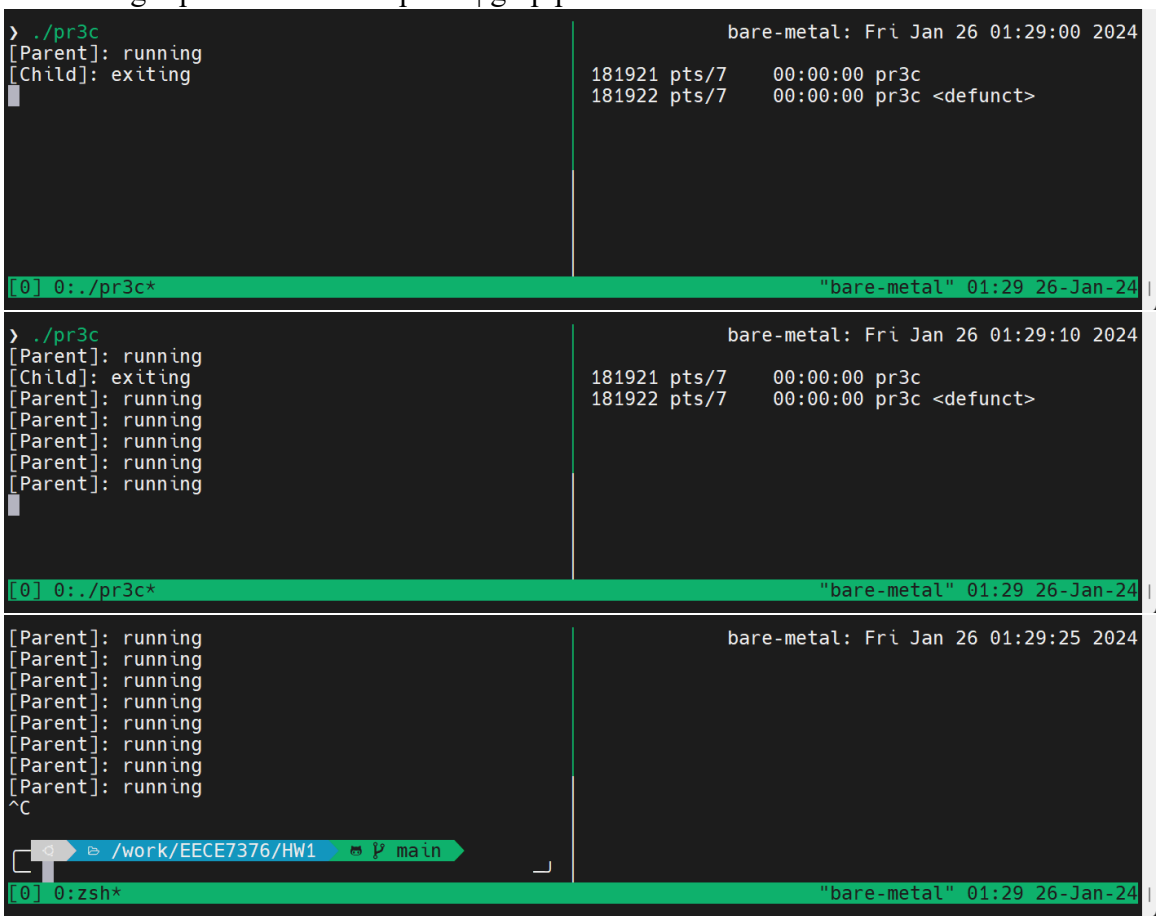
Problem 3c

Approach Summary:

The program creates a zombie process by having the child exit while the parent runs indefinitely, making the child a defunct process until the parent gets terminated.

Screenshot:

Right panel command: ps -A | grep pr



Problem 4

Approach Summary:

The get_args() function splits the string into arguments using strtok() and strdup(), and then prints each argument with the print_args() function.

Screenshot:

```
❯ ./pr4
Enter a string: wrsgvo weiosd efidcwaw8i09eida90 9eadu899u -ep 9efawdv
argv[0] = 'wrsgvo'
argv[1] = 'weiosd'
argv[2] = 'efidcwaw8i09eida90'
argv[3] = '9eadu899u'
argv[4] = '-ep'
argv[5] = '9efawdv'
❯ ./pr4
Enter a string: hello how are you
argv[0] = 'hello'
argv[1] = 'how'
argv[2] = 'are'
argv[3] = 'you'

/work/EECE7376/HW1  main

[0] 0:zsh*                                  "bare-metal" 01:39 26-Jan-24
```