A)Using Fork

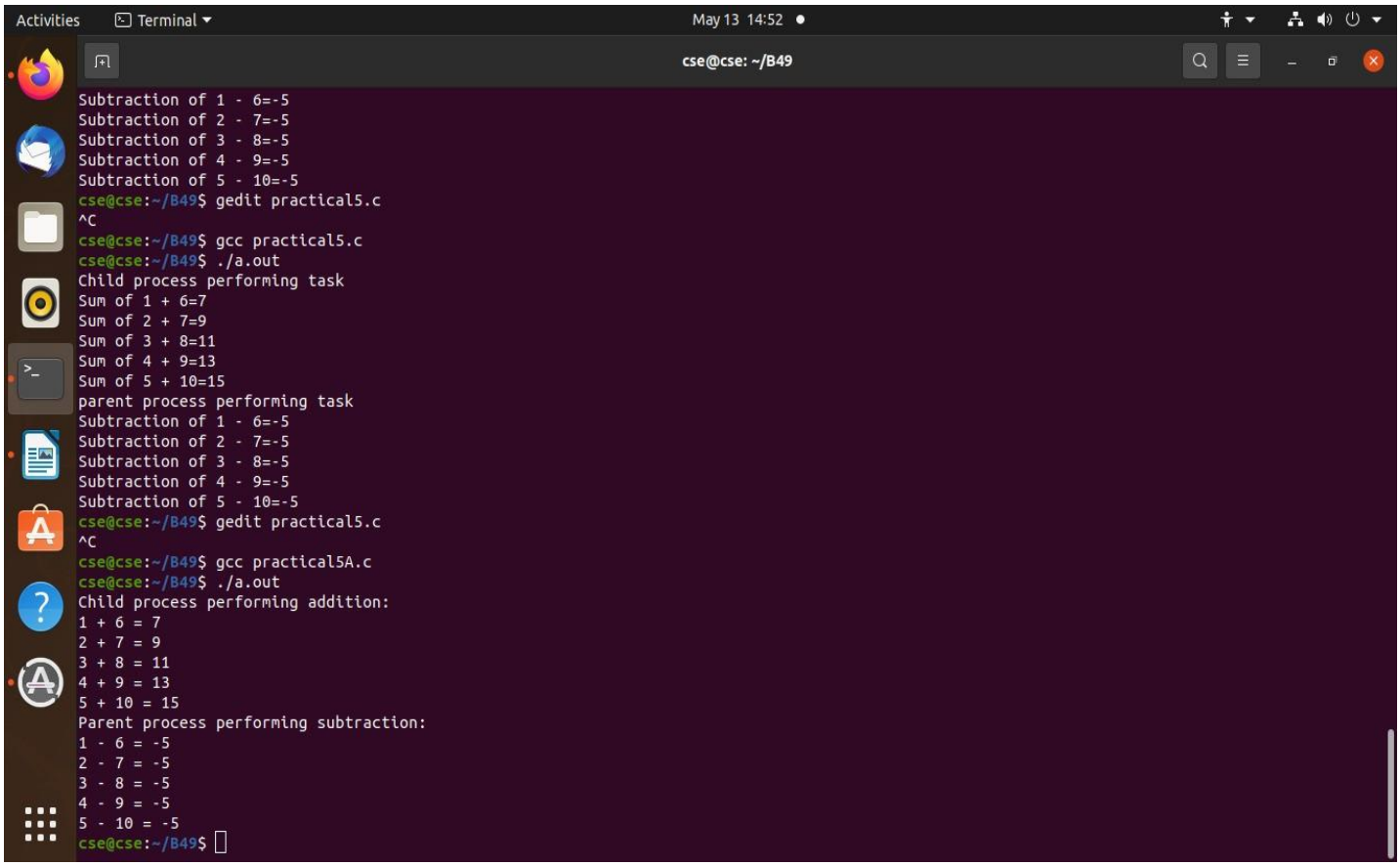
Code:

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

void performAddition(int array1[],int array2[],int size){
	printf("Child process performing task\n");
	for(int i=0; i<size; i++){
		int sum= array1[i]+array2[i];
		printf("Sum of %d +
%d=%d\n",array1[i],array2[i],sum);
	}
}

void performSubtraction(int array1[],int array2[],int size){
```

```
        printf("parent process performing task\n");
        for(int i=0; i<size; i++){
                int sub= array1[i]-array2[i];
                printf("Subtraction of %d -
%d=%d\n",array1[i],array2[i],sub);
        }
}

int main() {
    pid_t pid;
    int array1[] = {1, 2, 3, 4, 5};
    int array2[] = {6, 7, 8, 9, 10};
    int size = sizeof(array1) / sizeof(array1[0]);

    pid = fork();

    if (pid == 0) {
        sleep(5);
        performAddition(array1, array2, size);
        printf("Getpid()=%d\n",getpid());
    }
    else if (pid > 0) {
```

```
        wait(NULL);
        performSubtraction(array1, array2, size);
        printf("Getppid()=%d\n",getppid());
    } else {

        printf("Fork failed.\n");
        return 1;
    }

    return 0;
}
```

Output:



B) Using vfork

Code:

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

void performAddition(int array1[], int array2[], int size) {
    printf("Child process performing addition:\n");
    for (int i = 0; i < size; i++) {
        int result = array1[i] + array2[i];
        printf("%d + %d = %d\n", array1[i], array2[i], result);
    }
}

void performSubtraction(int array1[], int array2[], int size) {
    printf("Parent process performing subtraction:\n");
    for (int i = 0; i < size; i++) {
        int result = array1[i] - array2[i];
        printf("%d - %d = %d\n", array1[i], array2[i], result);
    }
```

```
}

int main() {
    pid_t pid;
    int array1[] = {1, 2, 3, 4, 5};
    int array2[] = {6, 7, 8, 9, 10};
    int size = sizeof(array1) / sizeof(array1[0]);

    pid = vfork();

    if (pid == 0) {
        // Child process
        performAddition(array1, array2, size);
        _exit(0);
    } else if (pid > 0) {
        // Parent process
        wait(NULL);
        performSubtraction(array1, array2, size);

    } else {
        // Fork failed
        printf("Fork failed.\n");
        return 1;
```

```
    }

    return 0;
}
```

Output: