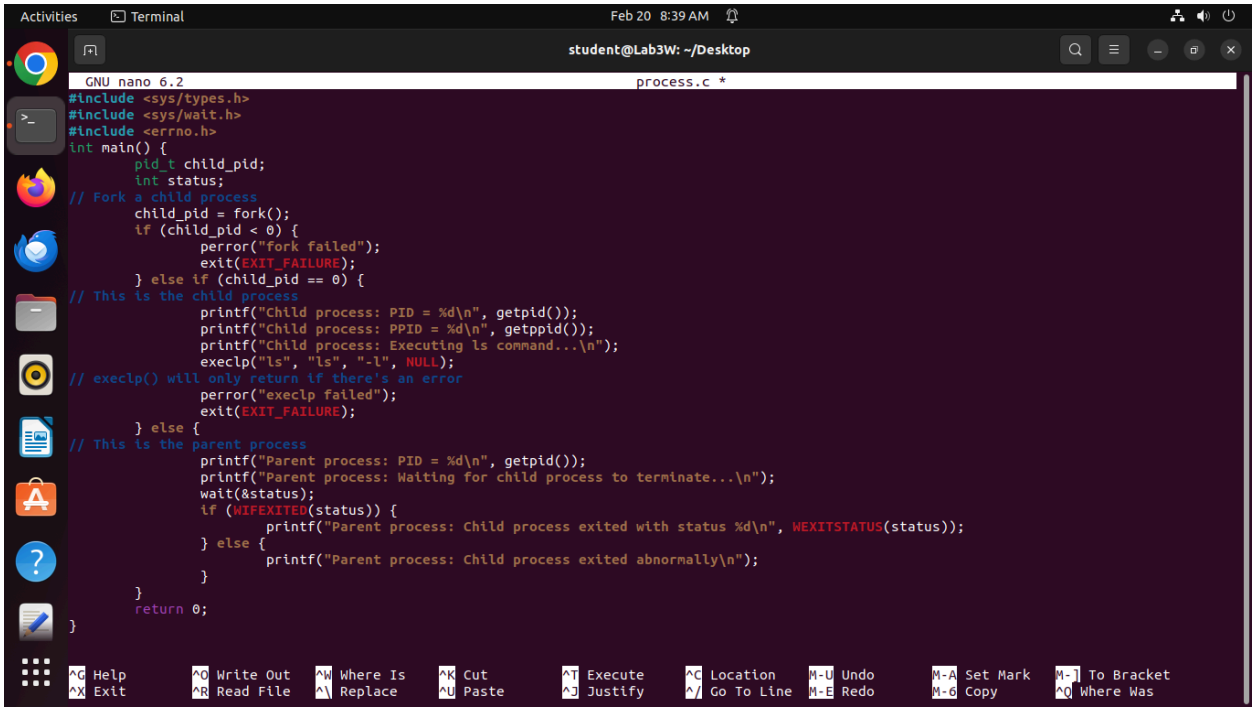
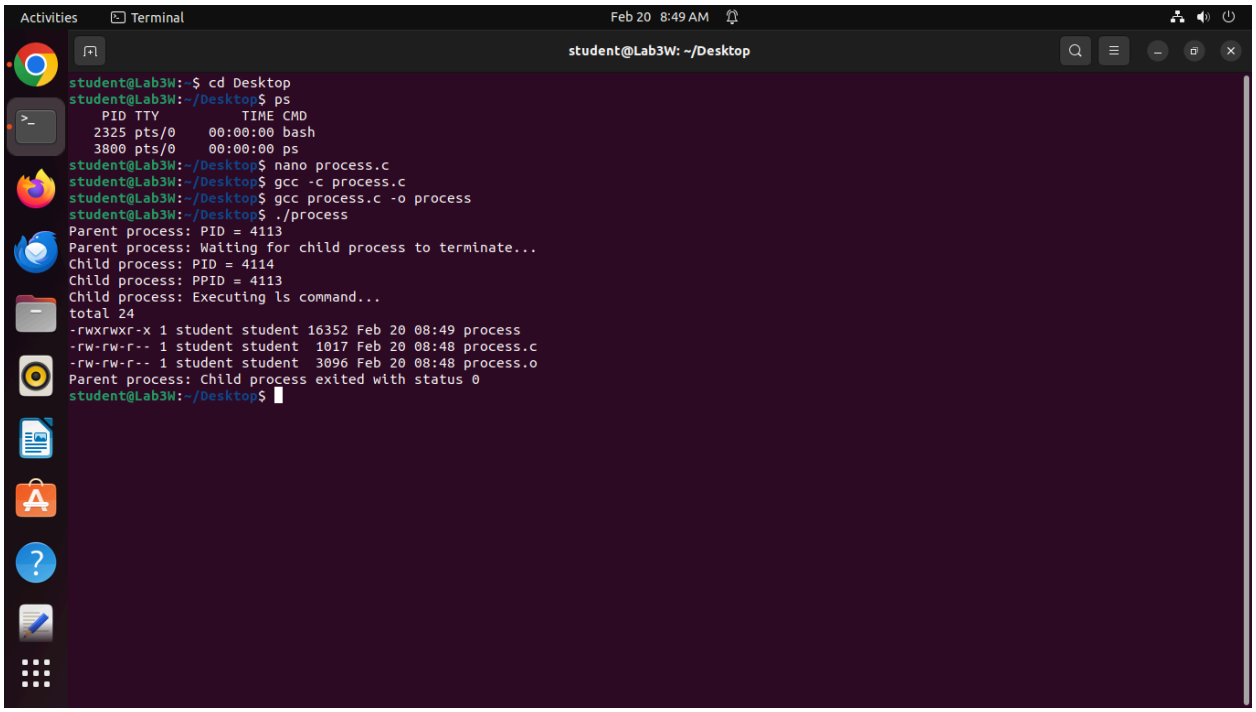
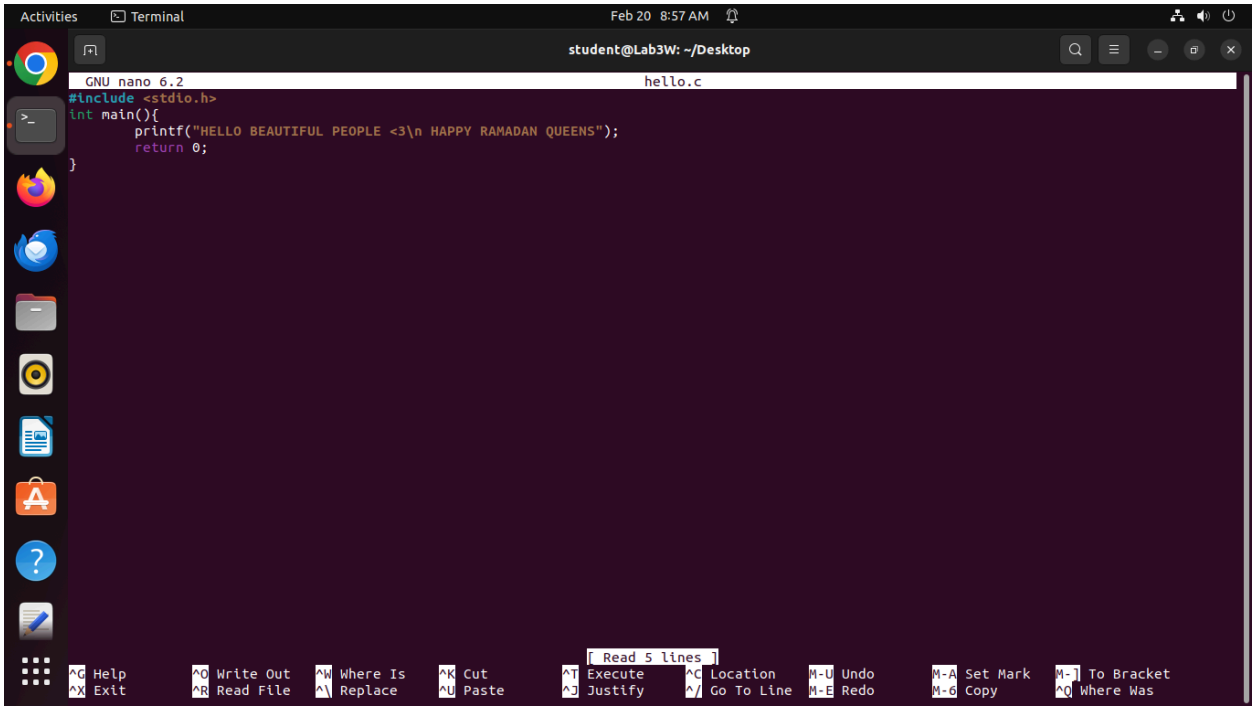
OS Lab 4:

24k-0810

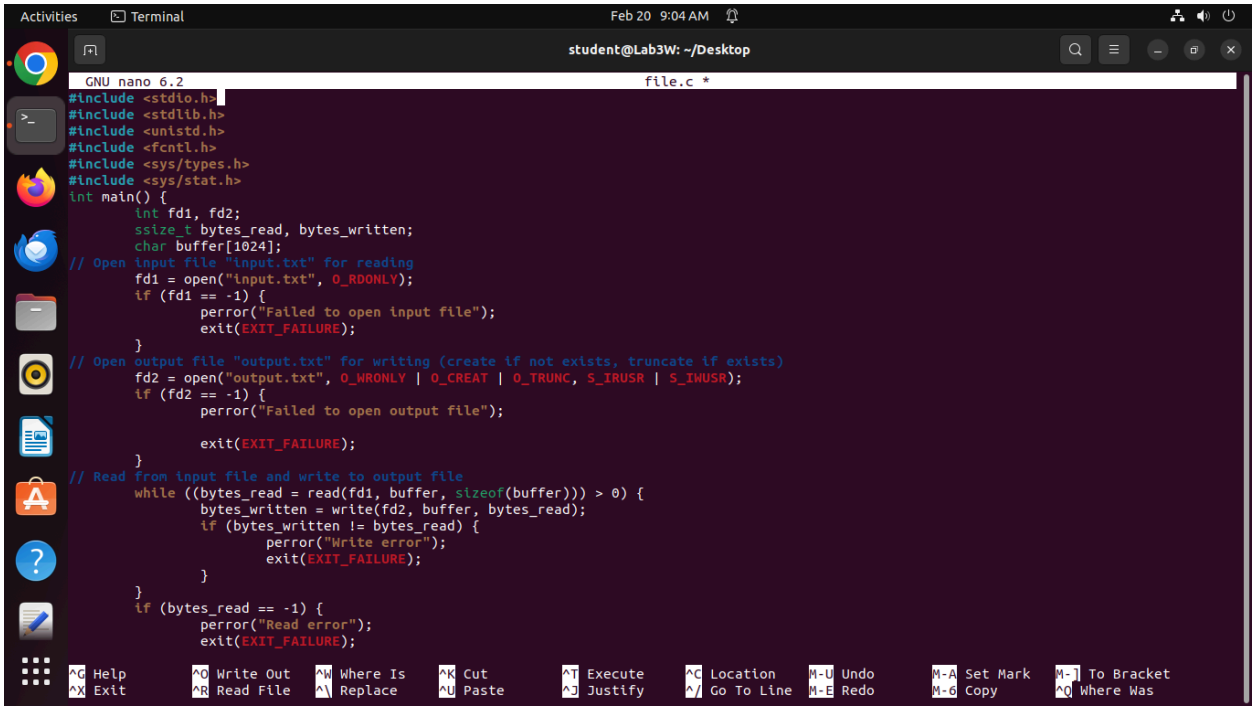
IN-LAB TASK

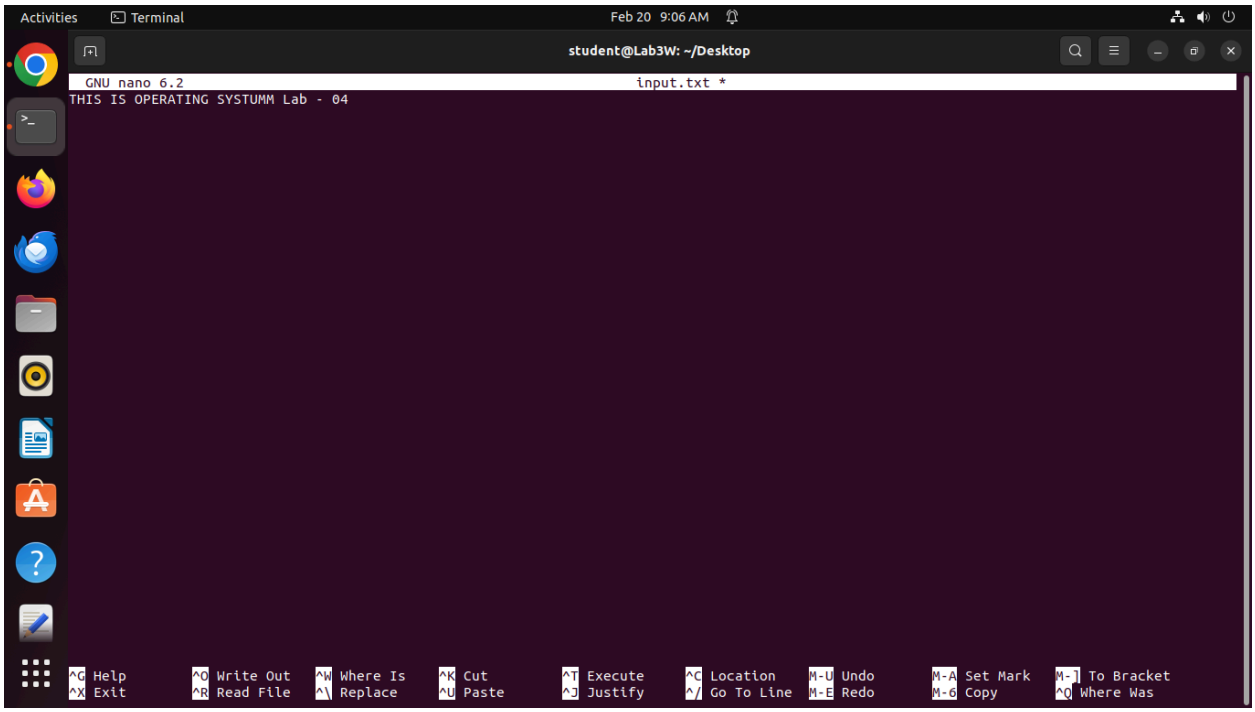


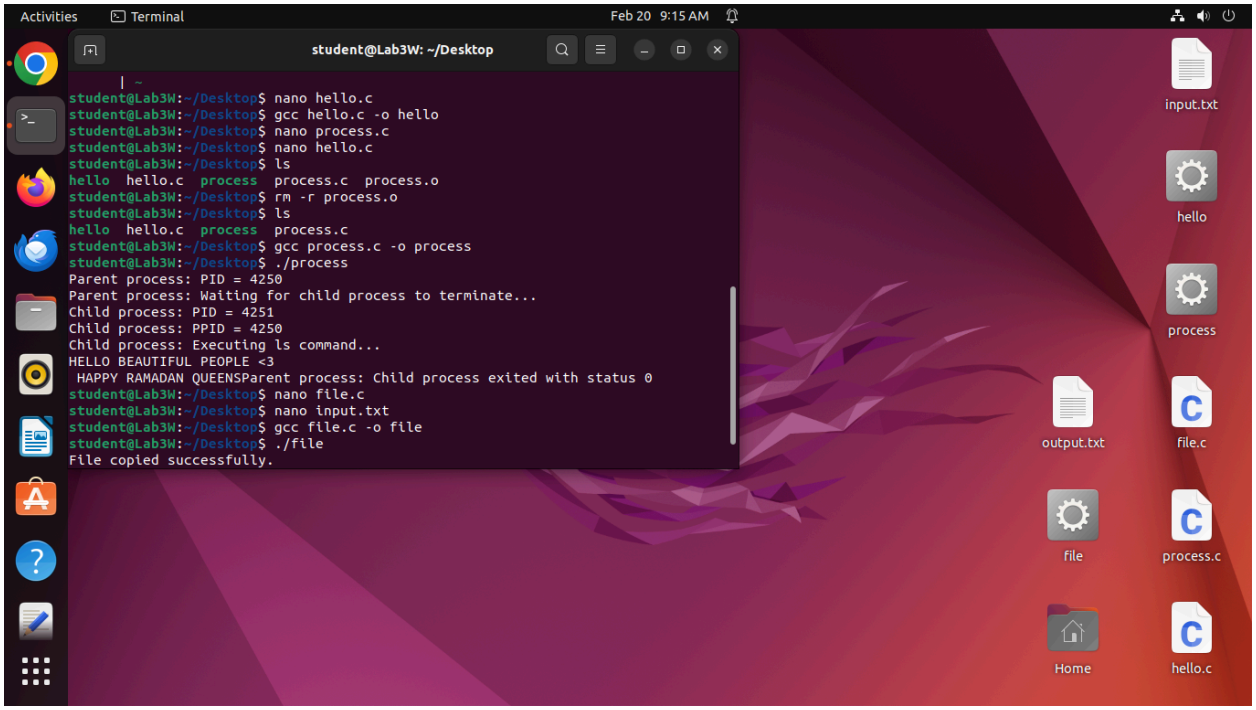



Activities
Terminal
Feb 20 8:57 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
hello.c
#include <stdio.h>
int main(){
printf("HELLO BEAUTIFUL PEOPLE <3\n HAPPY RAMADAN QUEENS");
return 0;
}
[Read 5 lines]
^G Help
^O Write Out
^W Where Is
^K Cut
^T Execute
^C Location
M-U Undo
M-A Set Mark
M-] To Bracket
^X Exit
^R Read File
^\ Replace
^U Paste
^J Justify
^/ Go To Line
M-E Redo
M-6 Copy
^Q Where Was

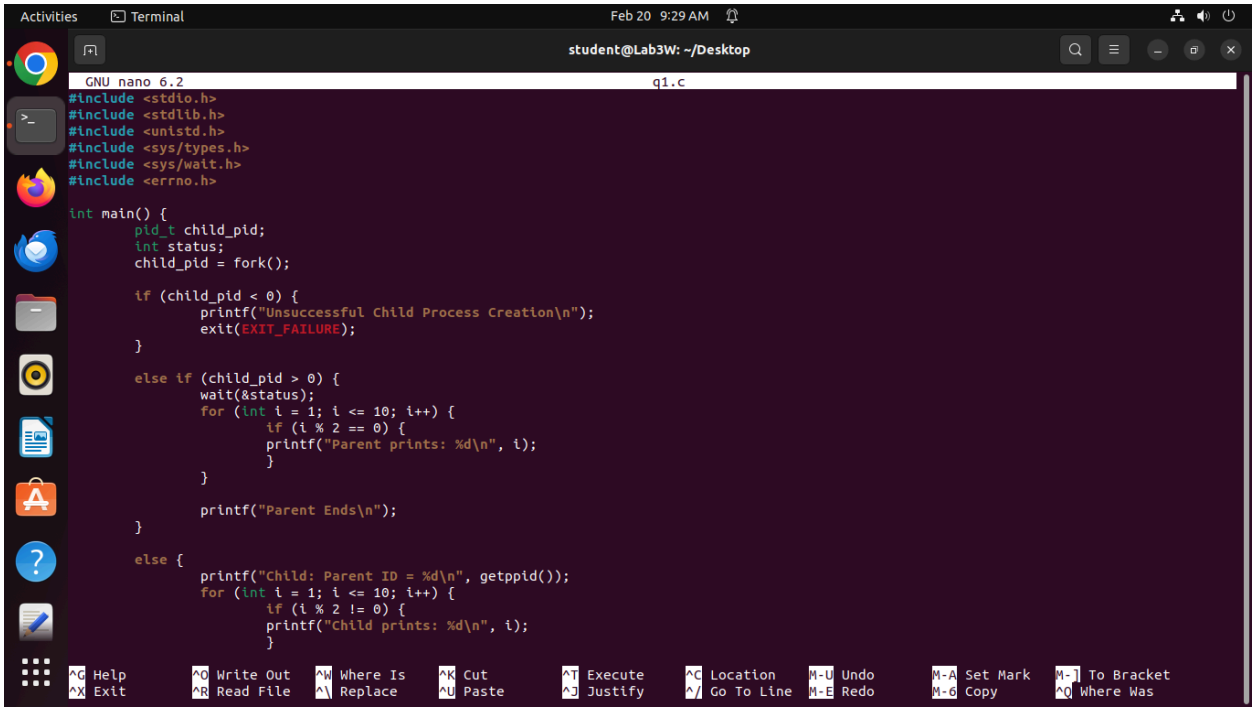

Activities
Terminal
Feb 20 8:56 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
process.c *
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <errno.h>
int main() {
pid_t child_pid;
int status;
// Fork a child process
child_pid = fork();
if (child_pid < 0) {
perror("fork failed");
exit(EXIT_FAILURE);
} else if (child_pid == 0) {
// This is the child process
printf("Child process: PID = %d\n", getpid());
printf("Child process: PPID = %d\n", getppid());
printf("Child process: Executing ls command...\n");
execlp("./hello", "hello", NULL);
// execlp() will only return if there's an error
perror("execlp failed");
exit(EXIT_FAILURE);
} else {
// This is the parent process
printf("Parent process: PID = %d\n", getpid());
printf("Parent process: Waiting for child process to terminate...\n");
wait(&status);
if (WIFEXITED(status)) {
printf("Parent process: Child process exited with status %d\n", WEXITSTATUS(status));
} else {
printf("Parent process: Child process exited abnormally\n");
}
}
Save modified buffer?
Y Yes
N No
^C Cancel


Activities
Terminal
Feb 20 9:04 AM
student@Lab3W: ~/Desktop
GNU nano 6.2 file.c *
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/stat.h>
int main() {
int fd1, fd2;
ssize_t bytes_read, bytes_written;
char buffer[1024];
// Open input file "input.txt" for reading
fd1 = open("input.txt", O_RDONLY);
if (fd1 == -1) {
perror("Failed to open input file");
exit(EXIT_FAILURE);
}
// Open output file "output.txt" for writing (create if not exists, truncate if exists)
fd2 = open("output.txt", O_WRONLY | O_CREAT | O_TRUNC, S_IRUSR | S_IWUSR);
if (fd2 == -1) {
perror("Failed to open output file");
exit(EXIT_FAILURE);
}
// Read from input file and write to output file
while ((bytes_read = read(fd1, buffer, sizeof(buffer))) > 0) {
bytes_written = write(fd2, buffer, bytes_read);
if (bytes_written != bytes_read) {
perror("Write error");
exit(EXIT_FAILURE);
}
}
if (bytes_read == -1) {
perror("Read error");
exit(EXIT_FAILURE);
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was

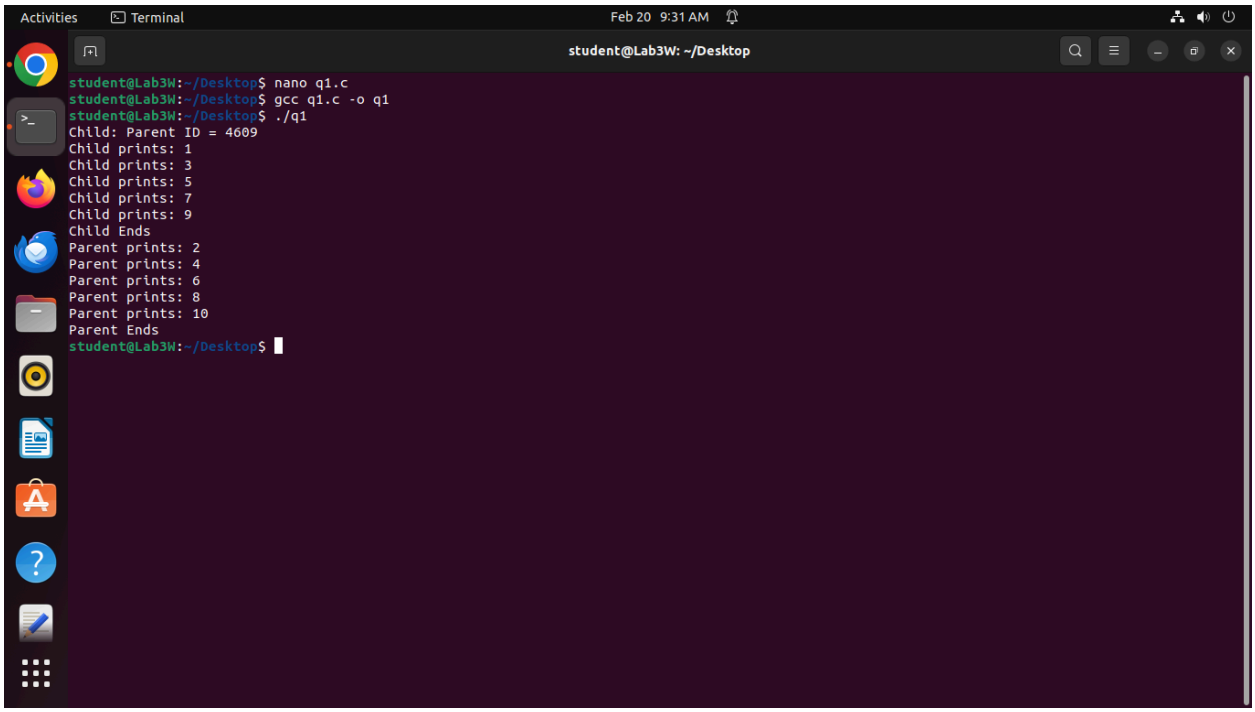

Activities
Terminal
Feb 20 9:06 AM
student@Lab3W: ~/Desktop
GNU nano 6.2 input.txt *
THIS IS OPERATING SYSTUMM Lab - 04
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was


Activities
Terminal
Feb 20 9:15 AM
student@Lab3W: ~/Desktop
student@Lab3W:~/Desktop$ nano hello.c
student@Lab3W:~/Desktop$ gcc hello.c -o hello
student@Lab3W:~/Desktop$ nano process.c
student@Lab3W:~/Desktop$ nano hello.c
student@Lab3W:~/Desktop$ ls
hello hello.c process process.c process.o
student@Lab3W:~/Desktop$ rm -r process.o
student@Lab3W:~/Desktop$ ls
hello hello.c process process.c
student@Lab3W:~/Desktop$ gcc process.c -o process
student@Lab3W:~/Desktop$ ./process
Parent process: PID = 4250
Parent process: Waiting for child process to terminate...
Child process: PID = 4251
Child process: PPID = 4250
Child process: Executing ls command...
HELLO BEAUTIFUL PEOPLE <3
HAPPY RAMADAN QUEENSParent process: Child process exited with status 0
student@Lab3W:~/Desktop$ nano file.c
student@Lab3W:~/Desktop$ nano input.txt
student@Lab3W:~/Desktop$ gcc file.c -o file
student@Lab3W:~/Desktop$ ./file
File copied successfully.
input.txt
hello
process
output.txt
file.c
file
process.c
Home
hello.c

Q1:

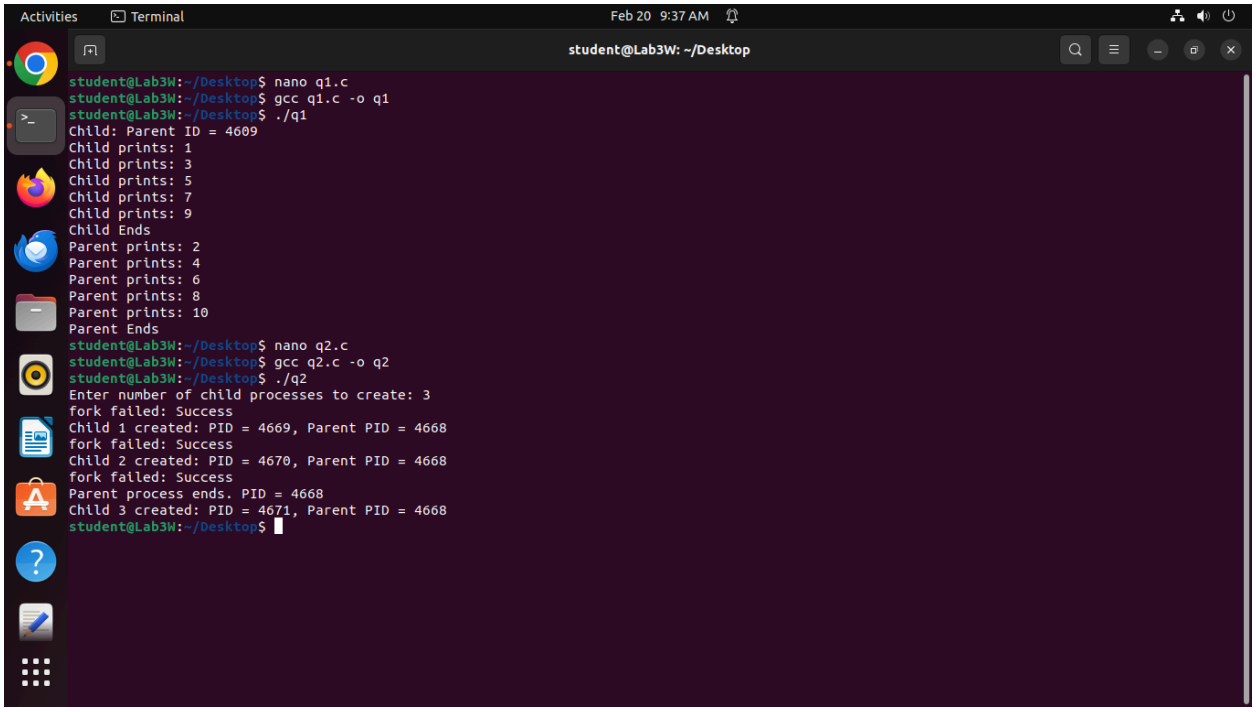

Activities
Terminal
Feb 20 9:29 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
q1.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <errno.h>
int main() {
pid_t child_pid;
int status;
child_pid = fork();
if (child_pid < 0) {
printf("Unsuccessful Child Process Creation\n");
exit(EXIT_FAILURE);
}
else if (child_pid > 0) {
wait(&status);
for (int i = 1; i <= 10; i++) {
if (i % 2 == 0) {
printf("Parent prints: %d\n", i);
}
}
printf("Parent Ends\n");
}
else {
printf("Child: Parent ID = %d\n", getppid());
for (int i = 1; i <= 10; i++) {
if (i % 2 != 0) {
printf("Child prints: %d\n", i);
}
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was


Activities
Terminal
Feb 20 9:29 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
q1.c
int main() {
pid_t child_pid;
int status;
child_pid = fork();
if (child_pid < 0) {
printf("Unsuccessful Child Process Creation\n");
exit(EXIT_FAILURE);
}
else if (child_pid > 0) {
wait(&status);
for (int i = 1; i <= 10; i++) {
if (i % 2 == 0) {
printf("Parent prints: %d\n", i);
}
}
printf("Parent Ends\n");
}
else {
printf("Child: Parent ID = %d\n", getppid());
for (int i = 1; i <= 10; i++) {
if (i % 2 != 0) {
printf("Child prints: %d\n", i);
}
}
printf("Child Ends\n");
}
return 0;
}
^G Help
^O Write Out
^W Where Is
^K Cut
^T Execute
^C Location
M-U Undo
M-A Set Mark
M-] To Bracket
^X Exit
^R Read File
^\ Replace
^U Paste
^J Justify
^/ Go To Line
M-E Redo
M-6 Copy
^Q Where Was


Activities
Terminal
Feb 20 9:31 AM
student@Lab3W: ~/Desktop
student@Lab3W:~/Desktop$ nano q1.c
student@Lab3W:~/Desktop$ gcc q1.c -o q1
student@Lab3W:~/Desktop$ ./q1
Child: Parent ID = 4609
Child prints: 1
Child prints: 3
Child prints: 5
Child prints: 7
Child prints: 9
Child Ends
Parent prints: 2
Parent prints: 4
Parent prints: 6
Parent prints: 8
Parent prints: 10
Parent Ends
student@Lab3W:~/Desktop$

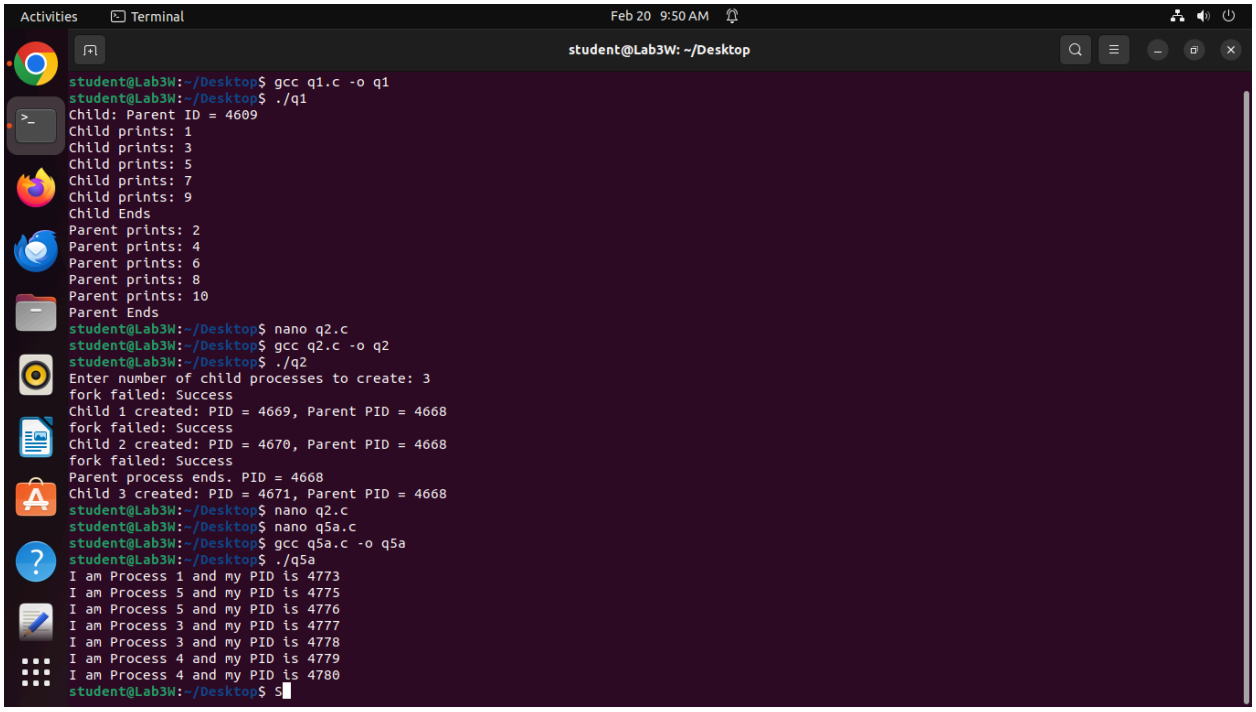
Q2:


Activities
Terminal
Feb 20 9:37 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
q2.c *
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <errno.h>
int main() {
int n;
printf("Enter number of child processes to create: ");
scanf("%d", &n);
for (int i = 0; i < n; i++) {
pid_t pid = fork();
if (pid < 0) {
printf("Fork failed\n");
exit(EXIT_FAILURE);
}
else if (pid == 0) {
printf("Child %d created: PID = %d, Parent PID = %d\n", i + 1, getpid(), getppid());
exit(0);
}
else{
perror("fork failed");
}
}
printf("Parent process ends. PID = %d\n", getpid());
return 0;
}
^G Help
^X Exit
^O Write Out
^R Read File
^W Where Is
^\ Replace
^K Cut
^U Paste
^T Execute
^J Justify
^C Location
^/ Go To Line
M-U Undo
M-E Redo
M-A Set Mark
M-6 Copy
M-] To Bracket
^Q Where Was


Activities
Terminal
Feb 20 9:37 AM
student@Lab3W: ~/Desktop
student@Lab3W:~/Desktop$ nano q1.c
student@Lab3W:~/Desktop$ gcc q1.c -o q1
student@Lab3W:~/Desktop$ ./q1
Child: Parent ID = 4609
Child prints: 1
Child prints: 3
Child prints: 5
Child prints: 7
Child prints: 9
Child Ends
Parent prints: 2
Parent prints: 4
Parent prints: 6
Parent prints: 8
Parent prints: 10
Parent Ends
student@Lab3W:~/Desktop$ nano q2.c
student@Lab3W:~/Desktop$ gcc q2.c -o q2
student@Lab3W:~/Desktop$ ./q2
Enter number of child processes to create: 3
fork failed: Success
Child 1 created: PID = 4669, Parent PID = 4668
fork failed: Success
Child 2 created: PID = 4670, Parent PID = 4668
fork failed: Success
Parent process ends. PID = 4668
Child 3 created: PID = 4671, Parent PID = 4668
student@Lab3W:~/Desktop$

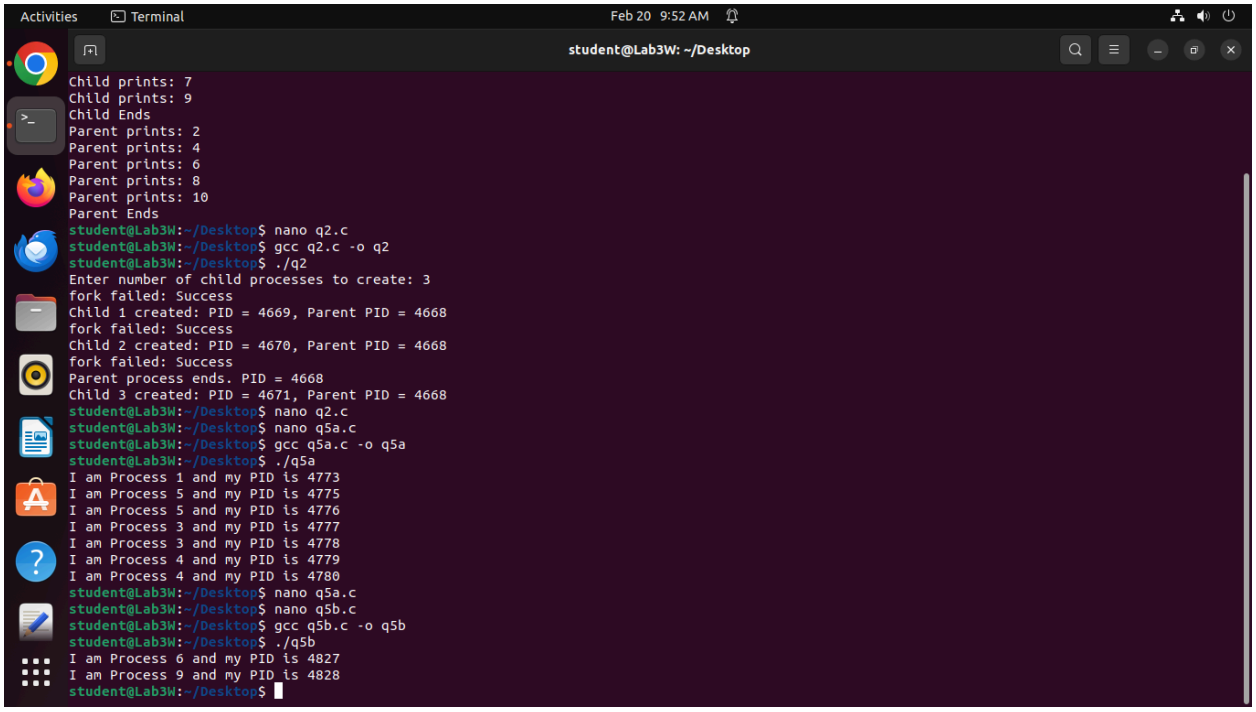
Q5a:


Activities
Terminal
Feb 20 9:50 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
q5a.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
int main(){
pid_t pid5, pid3, pid4;
printf("I am Process 1 and my PID is %d\n", getpid());
pid5=fork();
pid5 = fork();
if (pid5 < 0) {
perror("Fork failed");
exit(1);
}
if (pid5 == 0) {
printf("I am Process 5 and my PID is %d\n", getpid());
exit(0);
}
pid3 = fork();
if (pid3 < 0) {
perror("Fork failed");
exit(1);
}
if (pid3 == 0) {
printf("I am Process 3 and my PID is %d\n", getpid());
pid4 = fork();
if (pid4 < 0) {
perror("Fork failed");
exit(1);
}
if (pid4 == 0) {
[Read 44 lines]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was


Activities
Terminal
Feb 20 9:50 AM
student@Lab3W: ~/Desktop
student@Lab3W:~/Desktop$ gcc q1.c -o q1
student@Lab3W:~/Desktop$ ./q1
Child: Parent ID = 4609
Child prints: 1
Child prints: 3
Child prints: 5
Child prints: 7
Child prints: 9
Child Ends
Parent prints: 2
Parent prints: 4
Parent prints: 6
Parent prints: 8
Parent prints: 10
Parent Ends
student@Lab3W:~/Desktop$ nano q2.c
student@Lab3W:~/Desktop$ gcc q2.c -o q2
student@Lab3W:~/Desktop$ ./q2
Enter number of child processes to create: 3
fork failed: Success
Child 1 created: PID = 4669, Parent PID = 4668
fork failed: Success
Child 2 created: PID = 4670, Parent PID = 4668
fork failed: Success
Parent process ends. PID = 4668
Child 3 created: PID = 4671, Parent PID = 4668
student@Lab3W:~/Desktop$ nano q2.c
student@Lab3W:~/Desktop$ nano q5a.c
student@Lab3W:~/Desktop$ gcc q5a.c -o q5a
student@Lab3W:~/Desktop$ ./q5a
I am Process 1 and my PID is 4773
I am Process 5 and my PID is 4775
I am Process 5 and my PID is 4776
I am Process 3 and my PID is 4777
I am Process 3 and my PID is 4778
I am Process 4 and my PID is 4779
I am Process 4 and my PID is 4780
student@Lab3W:~/Desktop$ S

Q5b:


Activities
Terminal
Feb 20 9:51 AM
student@Lab3W: ~/Desktop
GNU nano 6.2
q5b.c *
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
int main() {
pid_t pid9;
printf("I am Process 6 and my PID is %d\n", getpid());
pid9 = fork();
if (pid9 < 0) {
perror("Fork failed");
exit(1);
}
if (pid9 == 0) {
printf("I am Process 9 and my PID is %d\n", getpid());
exit(0);
}
wait(NULL);
return 0;
}
^G Help
^X Exit
^O Write Out
^R Read File
^W Where Is
^\ Replace
^K Cut
^U Paste
^T Execute
^J Justify
^C Location
^/ Go To Line
M-U Undo
M-E Redo
M-A Set Mark
M-6 Copy
M-] To Bracket
^Q Where Was


Activities
Terminal
Feb 20 9:52 AM
student@Lab3W: ~/Desktop
Child prints: 7
Child prints: 9
Child Ends
Parent prints: 2
Parent prints: 4
Parent prints: 6
Parent prints: 8
Parent prints: 10
Parent Ends
student@Lab3W:~/Desktop$ nano q2.c
student@Lab3W:~/Desktop$ gcc q2.c -o q2
student@Lab3W:~/Desktop$ ./q2
Enter number of child processes to create: 3
fork failed: Success
Child 1 created: PID = 4669, Parent PID = 4668
fork failed: Success
Child 2 created: PID = 4670, Parent PID = 4668
fork failed: Success
Parent process ends. PID = 4668
Child 3 created: PID = 4671, Parent PID = 4668
student@Lab3W:~/Desktop$ nano q2.c
student@Lab3W:~/Desktop$ nano q5a.c
student@Lab3W:~/Desktop$ gcc q5a.c -o q5a
student@Lab3W:~/Desktop$ ./q5a
I am Process 1 and my PID is 4773
I am Process 5 and my PID is 4775
I am Process 5 and my PID is 4776
I am Process 3 and my PID is 4777
I am Process 3 and my PID is 4778
I am Process 4 and my PID is 4779
I am Process 4 and my PID is 4780
student@Lab3W:~/Desktop$ nano q5a.c
student@Lab3W:~/Desktop$ nano q5b.c
student@Lab3W:~/Desktop$ gcc q5b.c -o q5b
student@Lab3W:~/Desktop$ ./q5b
I am Process 6 and my PID is 4827
I am Process 9 and my PID is 4828
student@Lab3W:~/Desktop$