File Edit View Search Tools Documents Help

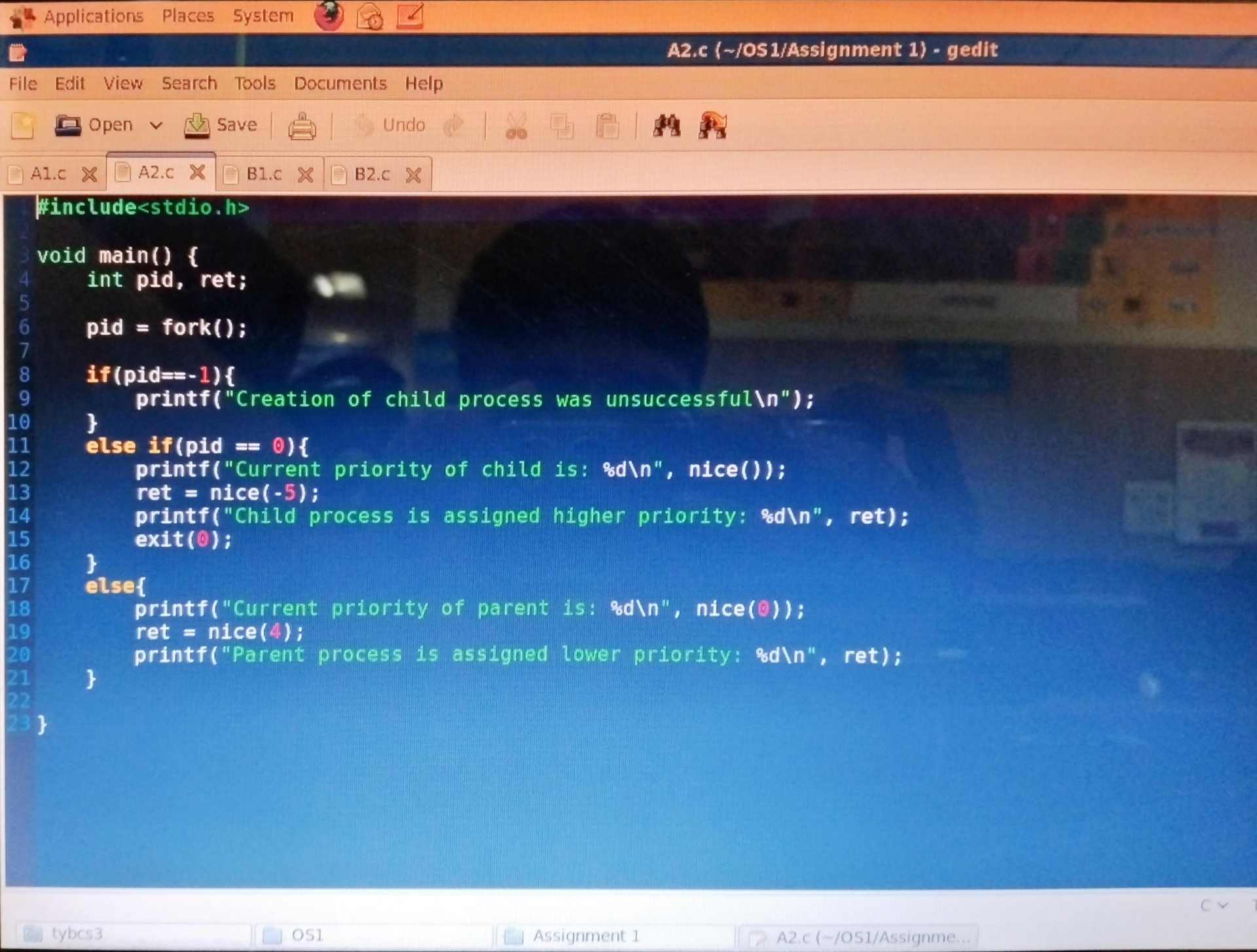
Open Save Undo

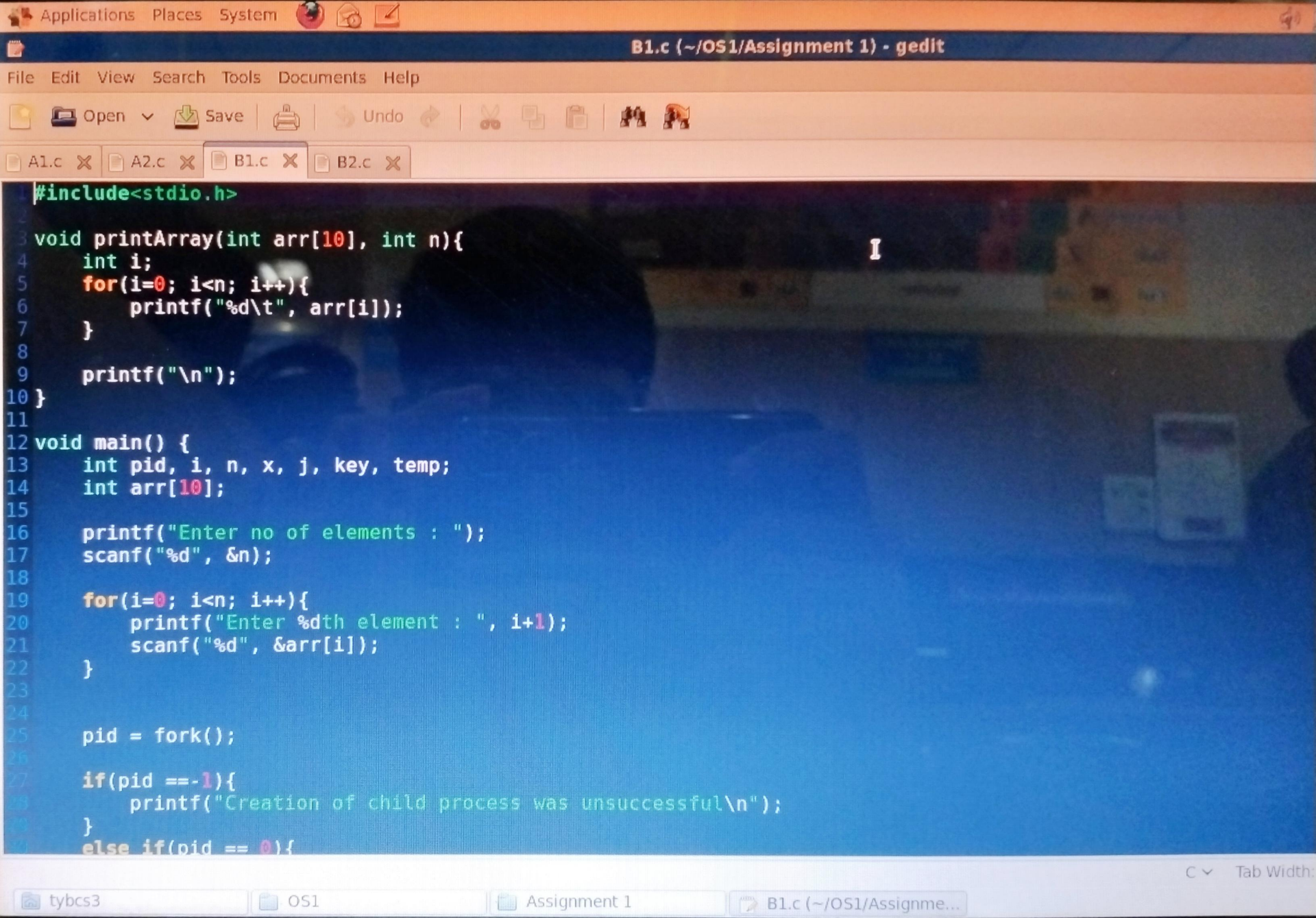
A1.c A2.c B1.c B2.c

```c
#include<stdio.h>

void main(){
    int i;
    i = fork();
    if(i==-1){
        printf("Creation of child process was unsuccessful\n");
    }
    else if(i==0){
        printf("Process id of child : %d\n", getpid());
        printf("I'm child process\n");
        exit(0);
    }
    else {
        printf("I'm parent process\n");
        printf("Process id of parent : %d\n", getpid());
    }
}
```

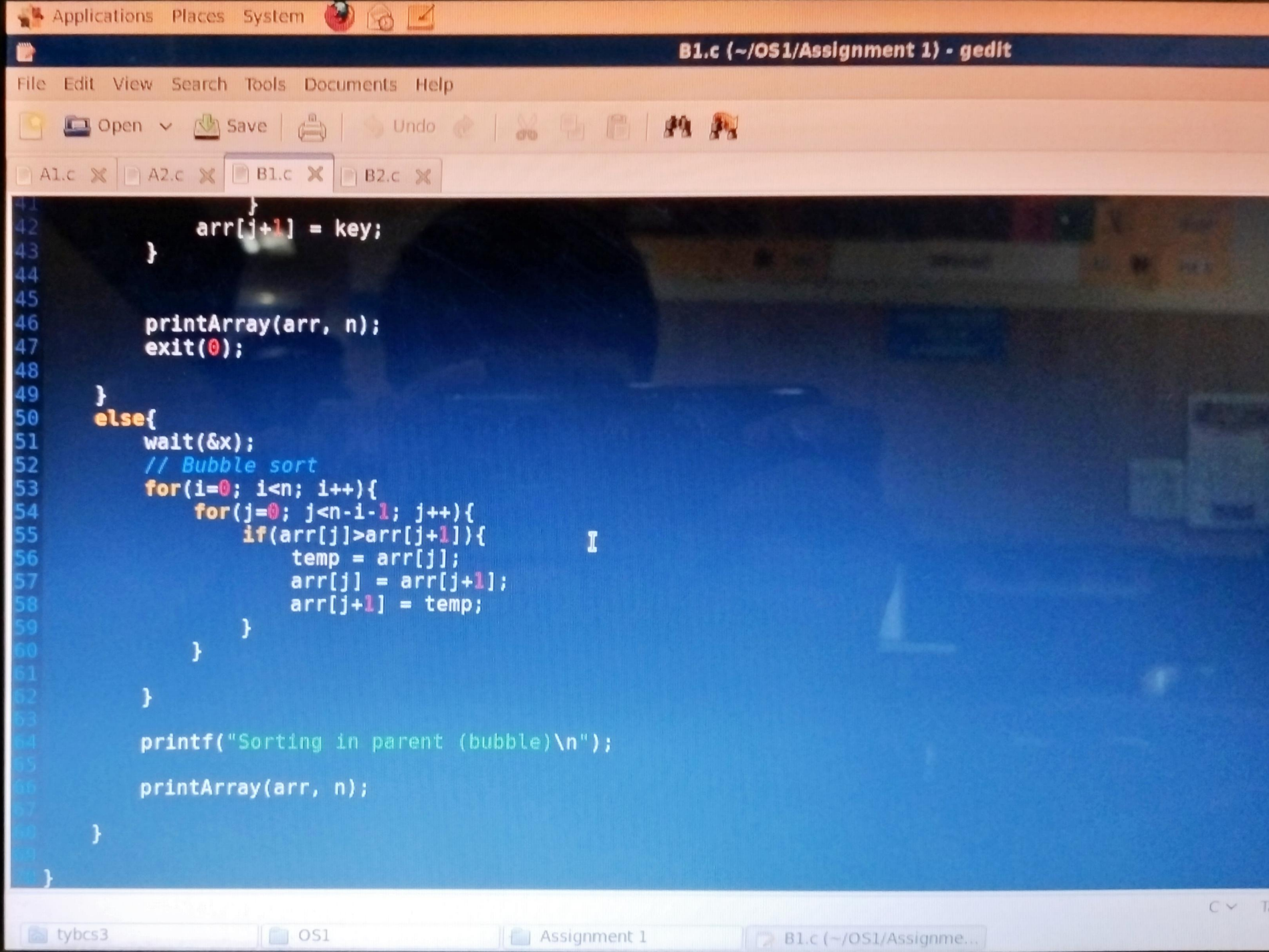
C Tab Width: 4 Ln 1,

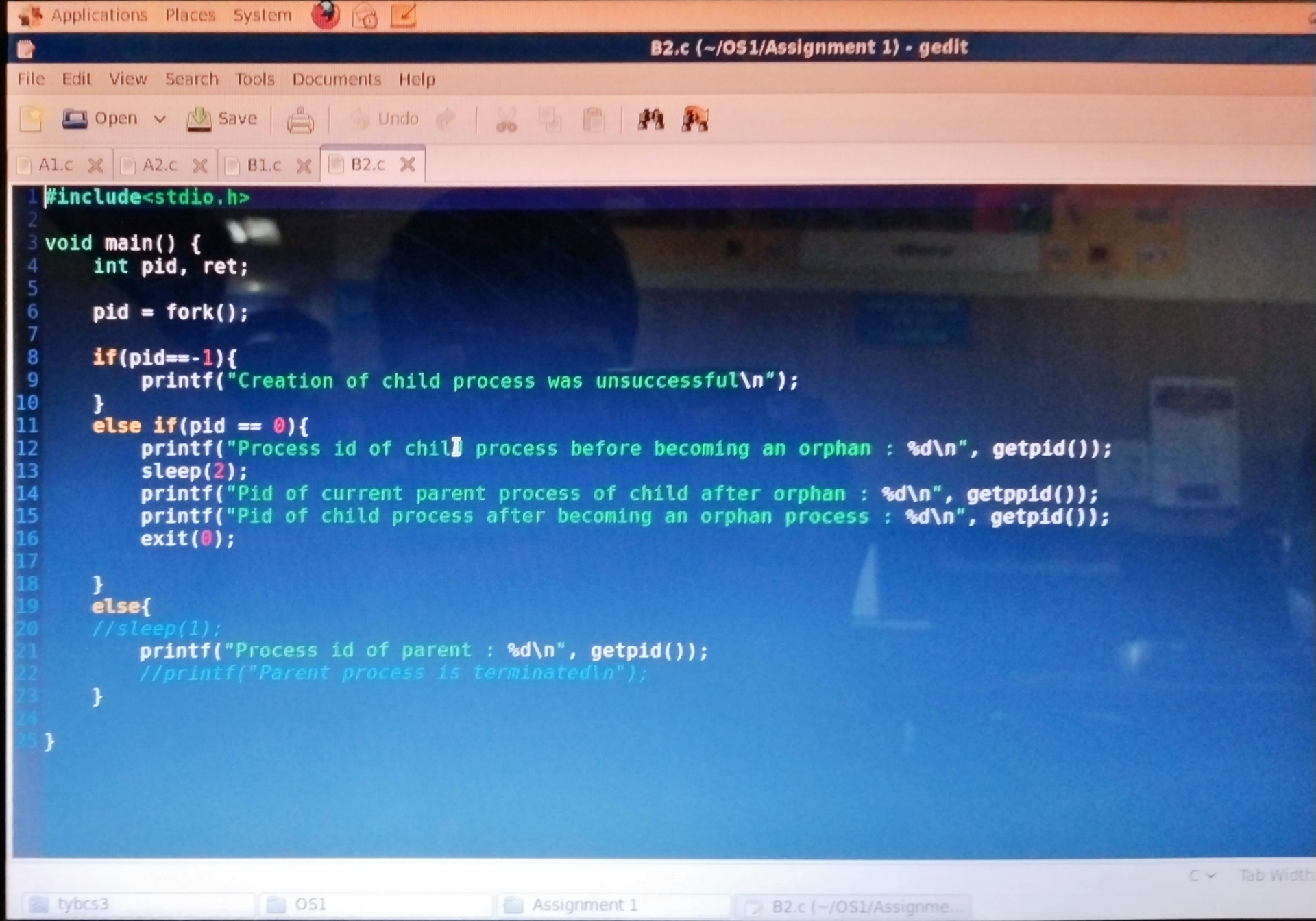
tybcs3 OS1 Assignment 1 A1.c (~/OS1/Assignme...


Applications Places System
A2.c (~/OS1/Assignment 1) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
A1.c A2.c B1.c B2.c
#include<stdio.h>
void main() {
int pid, ret;
pid = fork();
if(pid==-1){
printf("Creation of child process was unsuccessful\n");
}
else if(pid == 0){
printf("Current priority of child is: %d\n", nice());
ret = nice(-5);
printf("Child process is assigned higher priority: %d\n", ret);
exit(0);
}
else{
printf("Current priority of parent is: %d\n", nice(0));
ret = nice(4);
printf("Parent process is assigned lower priority: %d\n", ret);
}
}
tybcs3
OS1
Assignment 1
A2.c (~/OS1/Assignme...


Applications Places System
B1.c (~/OS1/Assignment 1) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
A1.c A2.c B1.c B2.c
#include<stdio.h>
void printArray(int arr[10], int n){
int i;
for(i=0; i<n; i++){
printf("%d\t", arr[i]);
}
printf("\n");
}
void main() {
int pid, i, n, x, j, key, temp;
int arr[10];
printf("Enter no of elements : ");
scanf("%d", &n);
for(i=0; i<n; i++){
printf("Enter %dth element : ", i+1);
scanf("%d", &arr[i]);
}
pid = fork();
if(pid ==-1){
printf("Creation of child process was unsuccessful\n");
}
else if(pid == 0){
C Tab Width:
tybcs3
OS1
Assignment 1
B1.c (~/OS1/Assignme...

```
    else if(pid == 0){
        // Insertion sort
        printf("Sorting in Child (insertion)\n");

        for(i=1; i<n; i++){
            key = arr[i];
            j= i-1;

                while(j>=0 && key<arr[j]){
                    arr[j+1] = arr[j];
                    j--;
                }
            arr[j+1] = key;
        }


        printArray(arr, n);
        exit(0);

    }
    else{
        wait(&x);
        // Bubble sort
        for(i=0; i<n; i++){
            for(j=0; j<n-i-1; j++){
                if(arr[j]>arr[j+1]){
                    temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
```


Applications Places System
B1.c (~/OS1/Assignment 1) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
A1.c A2.c B1.c B2.c
arr[j+1] = key;
}
printArray(arr, n);
exit(0);
}
else{
wait(&x);
// Bubble sort
for(i=0; i<n; i++){
for(j=0; j<n-i-1; j++){
if(arr[j]>arr[j+1]){
temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp;
}
}
}
printf("Sorting in parent (bubble)\n");
printArray(arr, n);
}
}
C
tybcs3
OS1
Assignment 1
B1.c (~/OS1/Assignme...


Applications Places System
B2.c (~/OS1/Assignment 1) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
A1.c A2.c B1.c B2.c
#include<stdio.h>
void main() {
int pid, ret;
pid = fork();
if(pid==-1){
printf("Creation of child process was unsuccessful\n");
}
else if(pid == 0){
printf("Process id of child process before becoming an orphan : %d\n", getpid());
sleep(2);
printf("Pid of current parent process of child after orphan : %d\n", getppid());
printf("Pid of child process after becoming an orphan process : %d\n", getpid());
exit(0);
}
else{
//sleep(1);
printf("Process id of parent : %d\n", getpid());
//printf("Parent process is terminated\n");
}
}
C Tab Width
tybcs3
OS1
Assignment 1
B2.c (~/OS1/Assignme...