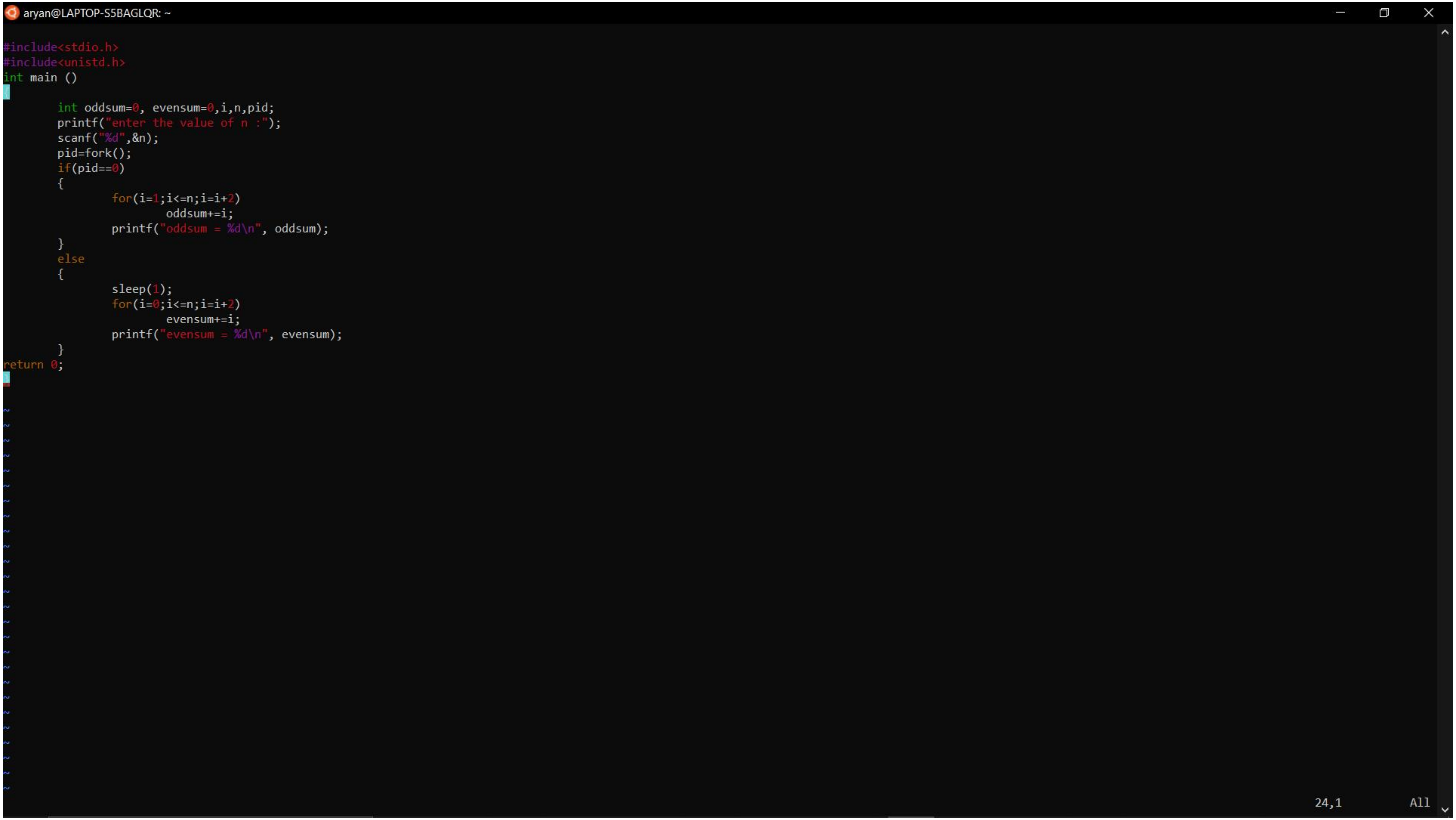
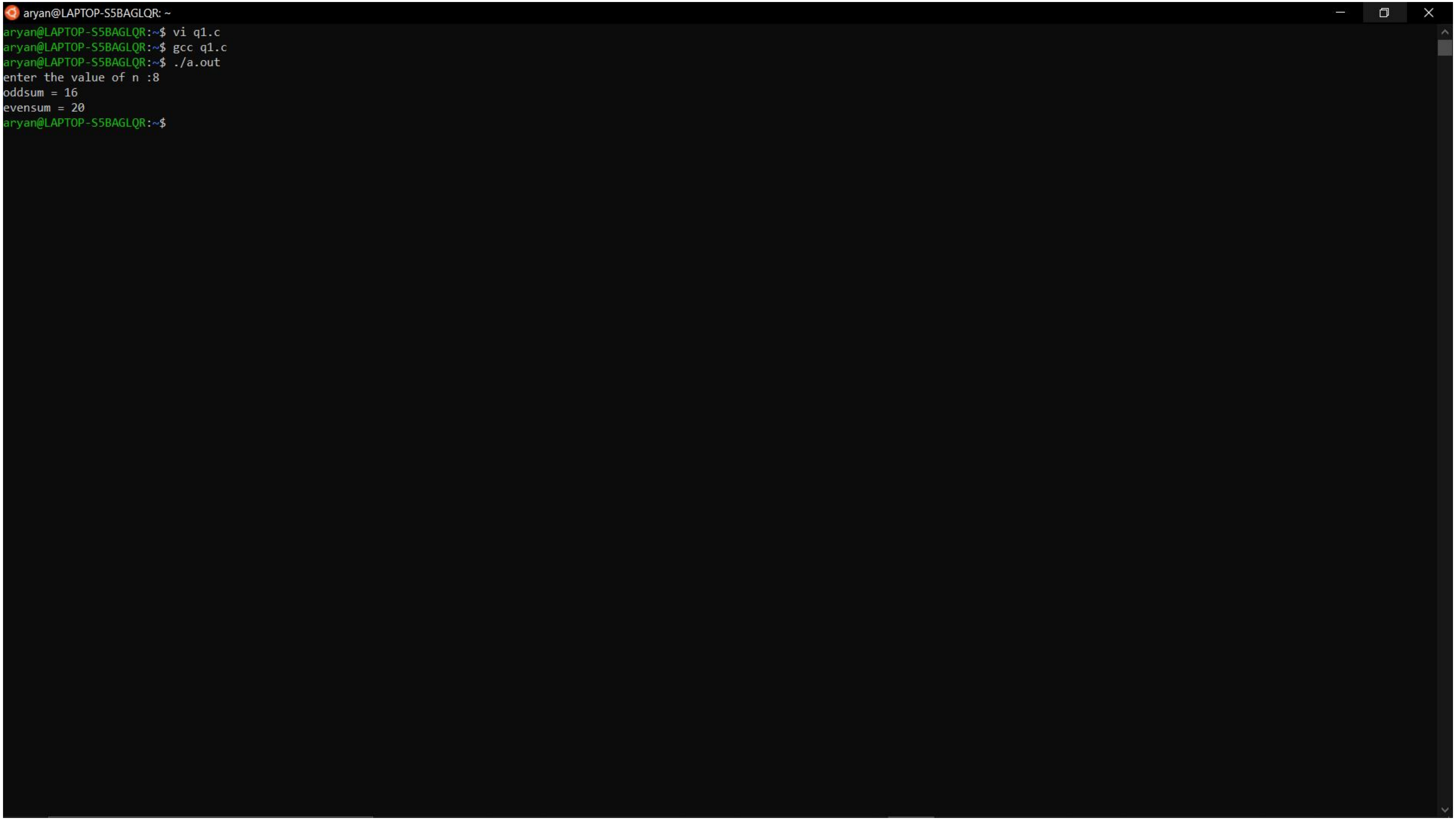
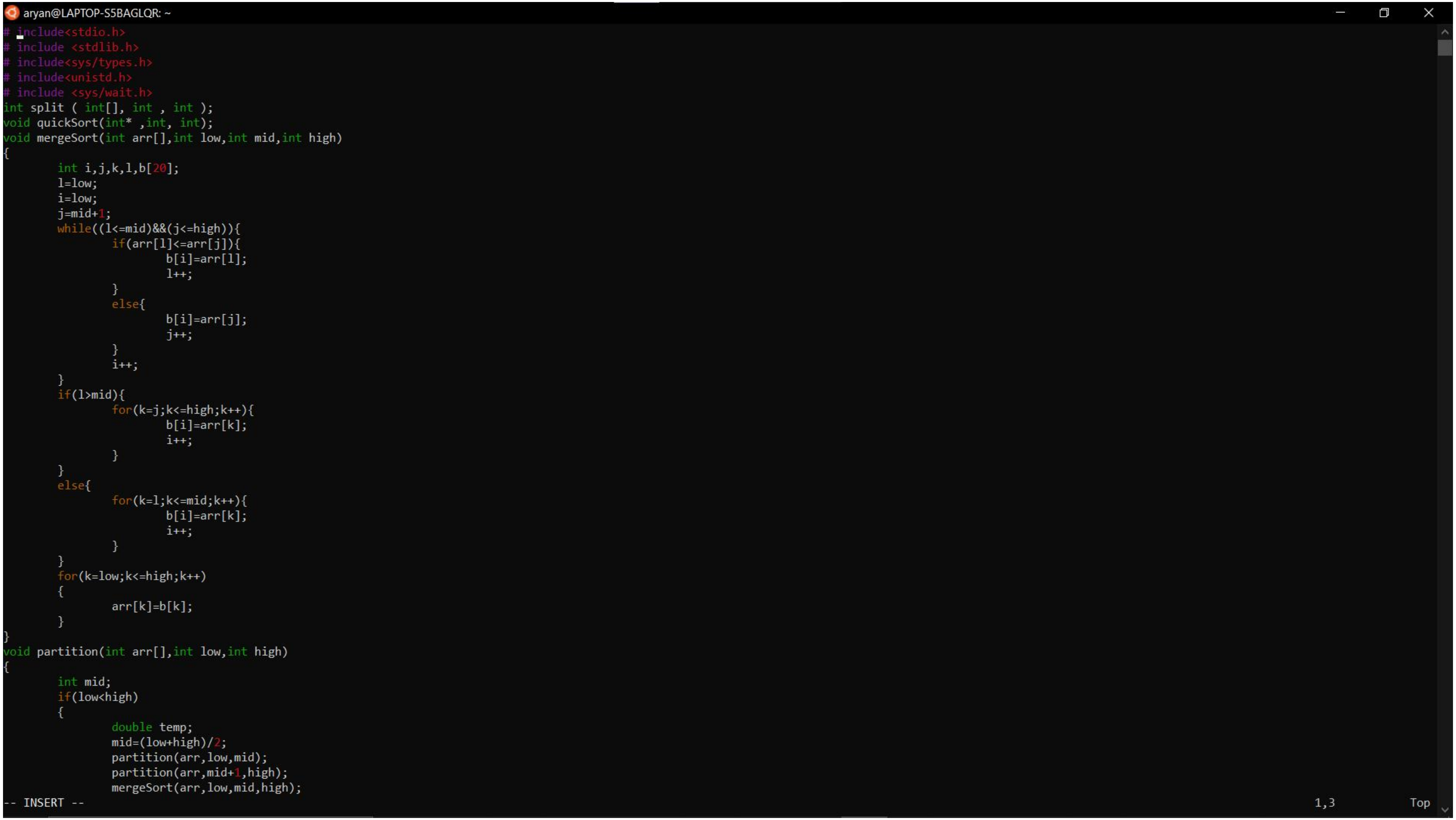


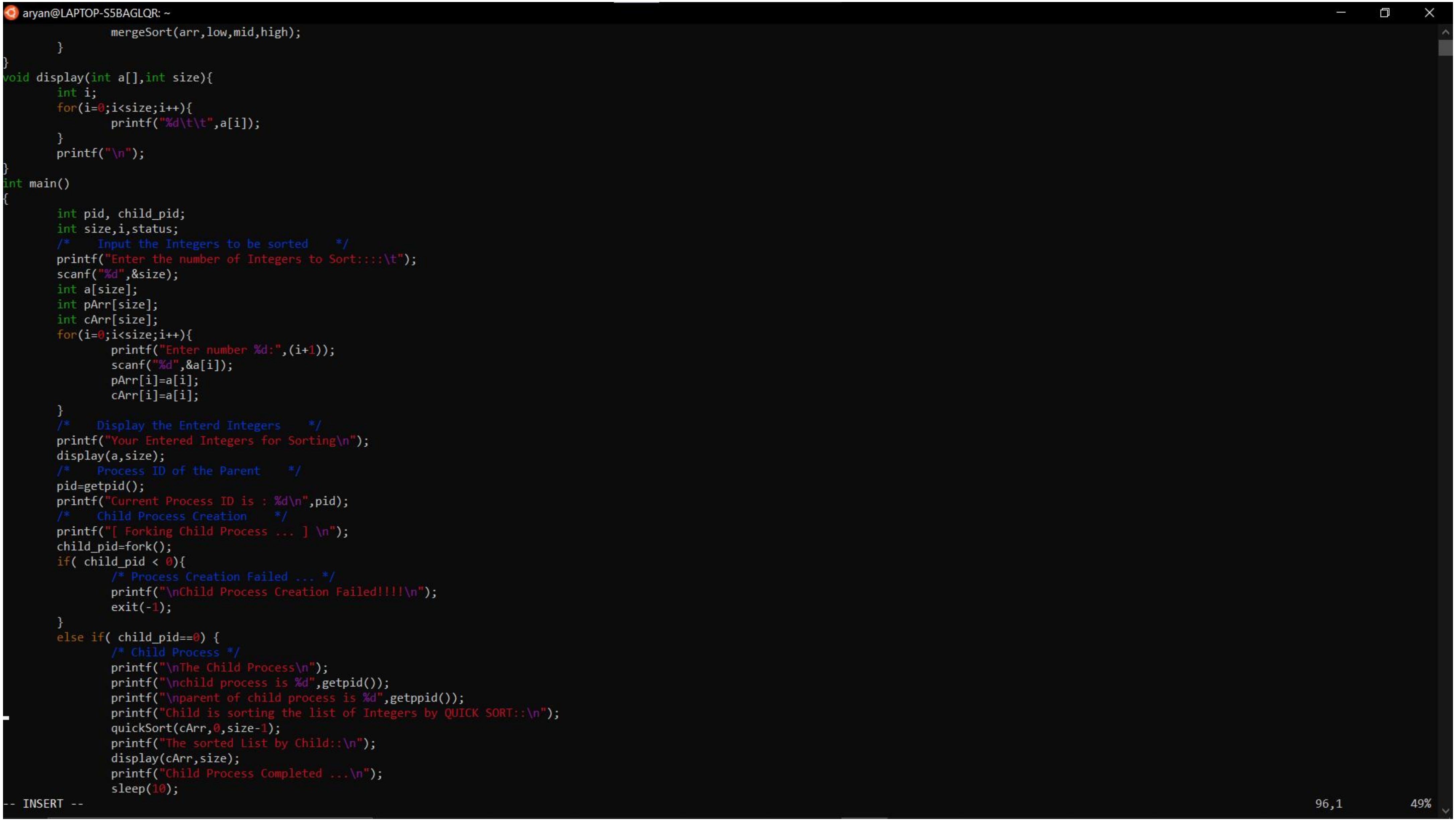
```c
#include<stdio.h>
#include<unistd.h>
int main ()
{
        int oddsum=0, evensum=0,i,n,pid;
        printf("enter the value of n :");
        scanf("%d",&n);
        pid=fork();
        if(pid==0)
        {
                for(i=1;i<=n;i=i+2)
                        oddsum+=i;
                printf("oddsum = %d\n", oddsum);
        }
        else
        {
                sleep(1);
                for(i=0;i<=n;i=i+2)
                        evensum+=i;
                printf("evensum = %d\n", evensum);
        }
return 0;
}
```



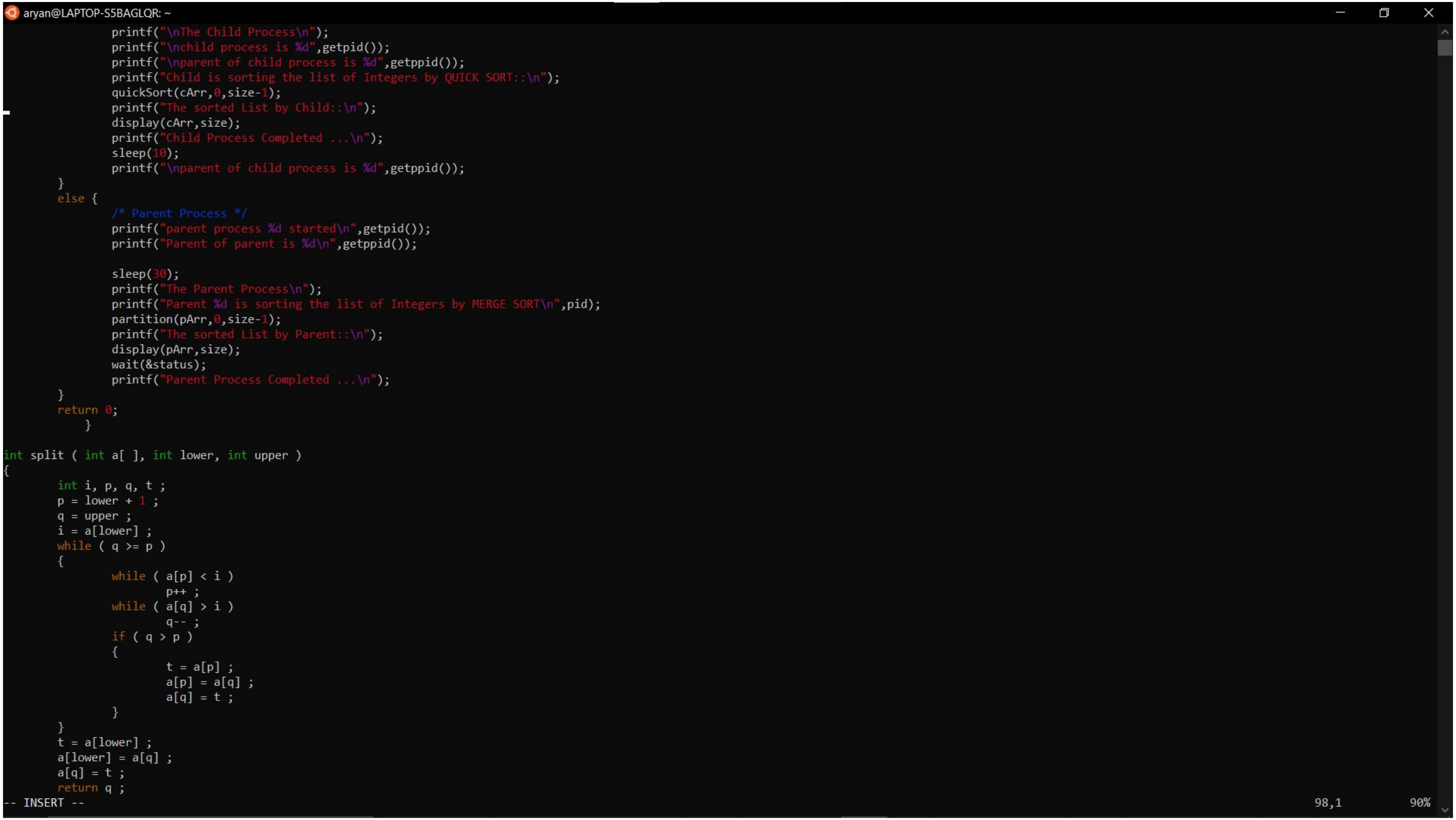
```
aryan@LAPTOP-S5BAGLQR:~$ vi q1.c
aryan@LAPTOP-S5BAGLQR:~$ gcc q1.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out
enter the value of n :8
oddsum = 16
evensum = 20
aryan@LAPTOP-S5BAGLQR:~$
```



```
# include<stdio.h>
# include <stdlib.h>
# include<sys/types.h>
# include<unistd.h>
# include <sys/wait.h>
int split ( int[], int , int );
void quickSort(int* ,int, int);
void mergeSort(int arr[],int low,int mid,int high)
{
        int i,j,k,l,b[20];
        l=low;
        i=low;
        j=mid+1;
        while((l<=mid)&&(j<=high)){
                if(arr[l]<=arr[j]){
                        b[i]=arr[l];
                        l++;
                }
                else{
                        b[i]=arr[j];
                        j++;
                }
                i++;
        }
        if(l>mid){
                for(k=j;k<=high;k++){
                        b[i]=arr[k];
                        i++;
                }
        }
        else{
                for(k=l;k<=mid;k++){
                        b[i]=arr[k];
                        i++;
                }
        }
        for(k=low;k<=high;k++)
        {
                arr[k]=b[k];
        }
}
void partition(int arr[],int low,int high)
{
        int mid;
        if(low<high)
        {
                double temp;
                mid=(low+high)/2;
                partition(arr,low,mid);
                partition(arr,mid+1,high);
                mergeSort(arr,low,mid,high);
-- INSERT --                                                                                                                    1,3           Top
```

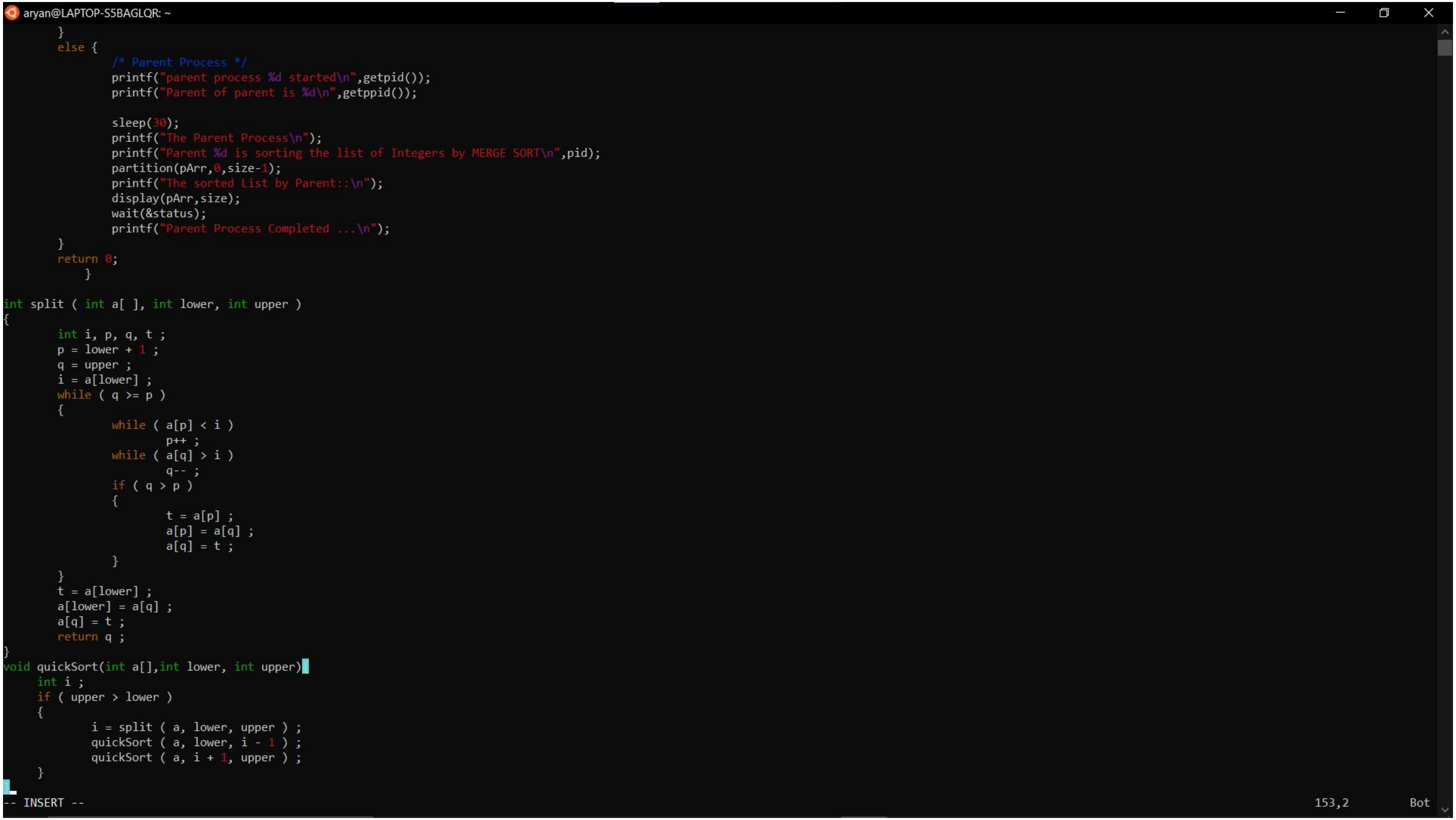


```
                mergeSort(arr,low,mid,high);
        }
}
void display(int a[],int size){
        int i;
        for(i=0;i<size;i++){
                printf("%d\t\t",a[i]);
        }
        printf("\n");
}
int main()
{
        int pid, child_pid;
        int size,i,status;
        /*    Input the Integers to be sorted    */
        printf("Enter the number of Integers to Sort::::\t");
        scanf("%d",&size);
        int a[size];
        int pArr[size];
        int cArr[size];
        for(i=0;i<size;i++){
                printf("Enter number %d:",(i+1));
                scanf("%d",&a[i]);
                pArr[i]=a[i];
                cArr[i]=a[i];
        }
        /*    Display the Enterd Integers    */
        printf("Your Entered Integers for Sorting\n");
        display(a,size);
        /*    Process ID of the Parent    */
        pid=getpid();
        printf("Current Process ID is : %d\n",pid);
        /*    Child Process Creation    */
        printf("[ Forking Child Process ... ] \n");
        child_pid=fork();
        if( child_pid < 0){
                /* Process Creation Failed ... */
                printf("\nChild Process Creation Failed!!!!\n");
                exit(-1);
        }
        else if( child_pid==0) {
                /* Child Process */
                printf("\nThe Child Process\n");
                printf("\nchild process is %d",getpid());
                printf("\nparent of child process is %d",getppid());
                printf("Child is sorting the list of Integers by QUICK SORT::\n");
                quickSort(cArr,0,size-1);
                printf("The sorted List by Child::\n");
                display(cArr,size);
                printf("Child Process Completed ...\n");
                sleep(10);
-- INSERT --                                                                                          96,1          49%
```

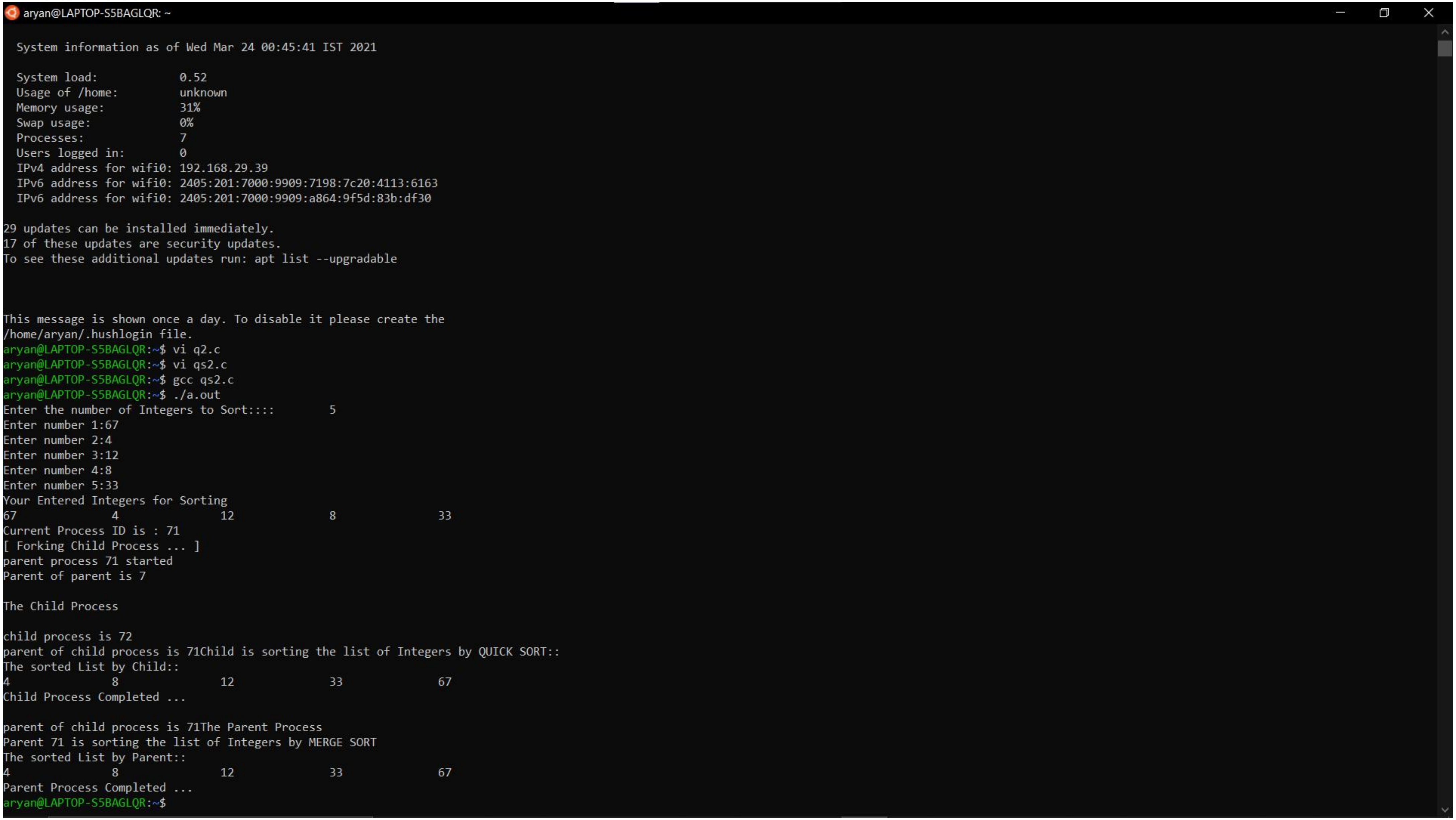


aryan@LAPTOP-S5BAGLQR: ~

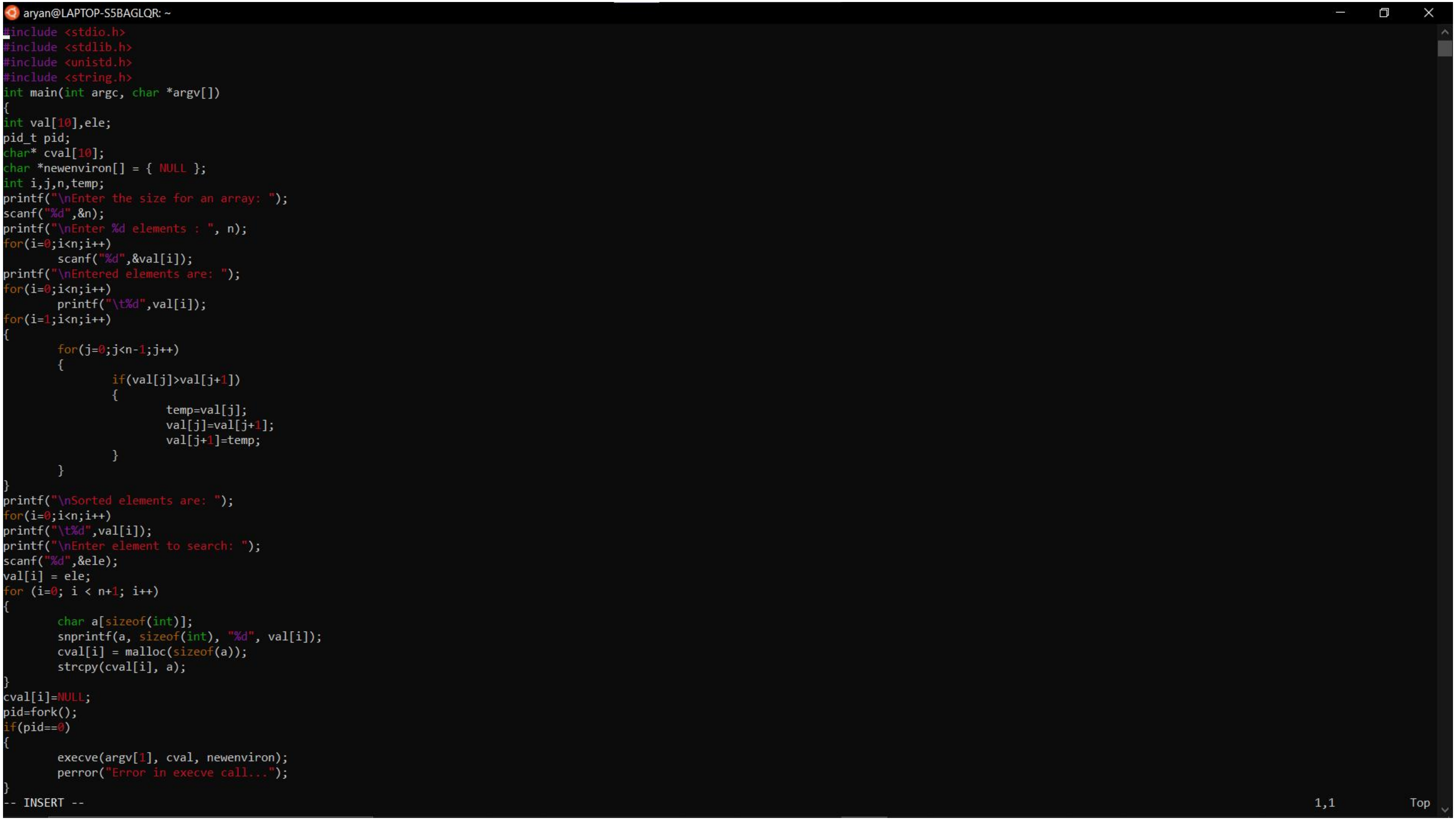
```
                printf("\nThe Child Process\n");
                printf("\nchild process is %d",getpid());
                printf("\nparent of child process is %d",getppid());
                printf("Child is sorting the list of Integers by QUICK SORT::\n");
                quickSort(cArr,0,size-1);
                printf("The sorted List by Child::\n");
                display(cArr,size);
                printf("Child Process Completed ...\n");
                sleep(10);
                printf("\nparent of child process is %d",getppid());
        }
        else {
                /* Parent Process */
                printf("parent process %d started\n",getpid());
                printf("Parent of parent is %d\n",getppid());

                sleep(30);
                printf("The Parent Process\n");
                printf("Parent %d is sorting the list of Integers by MERGE SORT\n",pid);
                partition(pArr,0,size-1);
                printf("The sorted List by Parent::\n");
                display(pArr,size);
                wait(&status);
                printf("Parent Process Completed ...\n");
        }
        return 0;
            }

int split ( int a[ ], int lower, int upper )
{
        int i, p, q, t ;
        p = lower + 1 ;
        q = upper ;
        i = a[lower] ;
        while ( q >= p )
        {
                while ( a[p] < i )
                        p++ ;
                while ( a[q] > i )
                        q-- ;
                if ( q > p )
                {
                        t = a[p] ;
                        a[p] = a[q] ;
                        a[q] = t ;
                }
        }
        t = a[lower] ;
        a[lower] = a[q] ;
        a[q] = t ;
        return q ;
-- INSERT --                                                                    98,1          90%
```



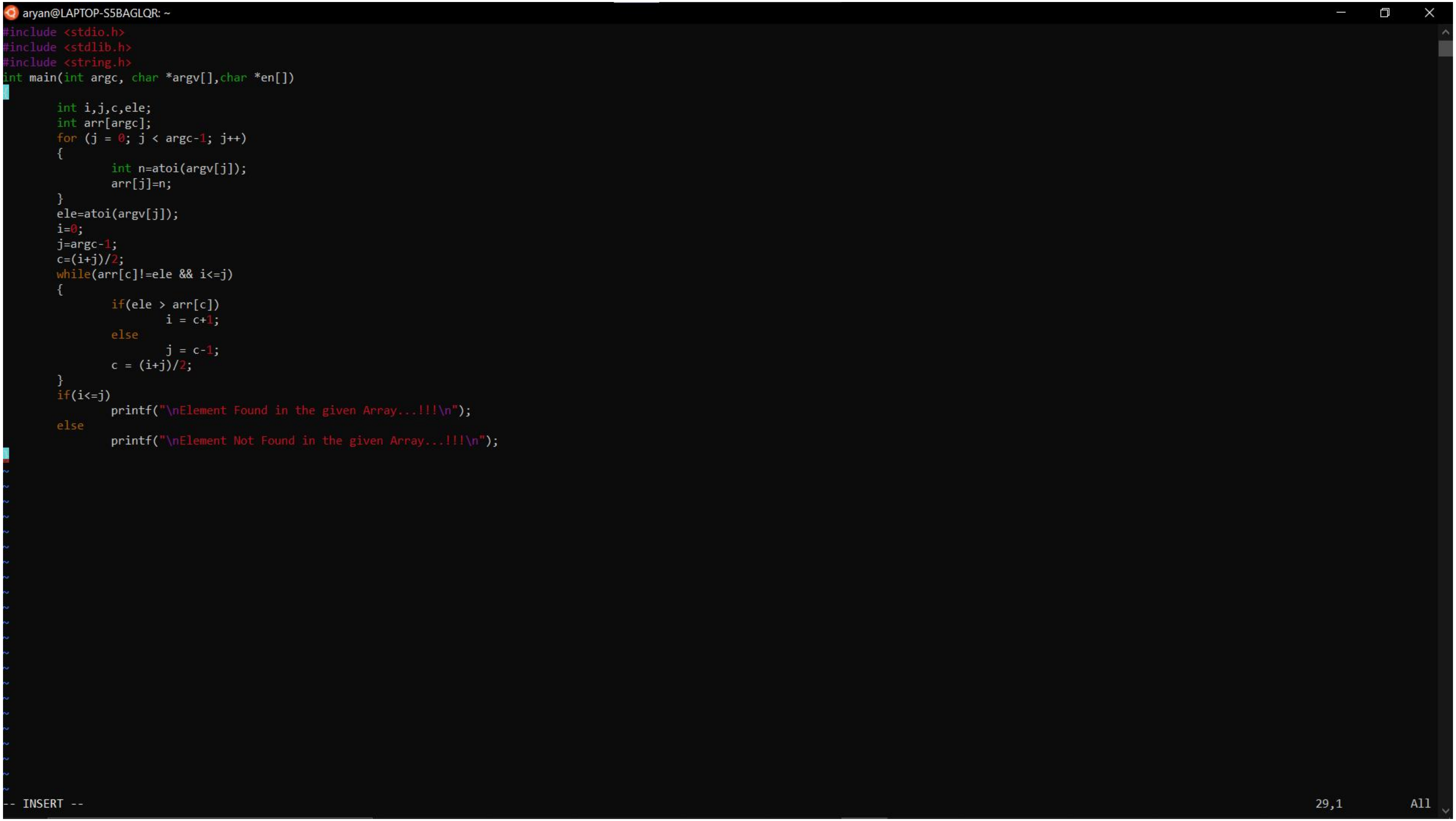
```
        }
        else {
                /* Parent Process */
                printf("parent process %d started\n",getpid());
                printf("Parent of parent is %d\n",getppid());

                sleep(30);
                printf("The Parent Process\n");
                printf("Parent %d is sorting the list of Integers by MERGE SORT\n",pid);
                partition(pArr,0,size-1);
                printf("The sorted List by Parent::\n");
                display(pArr,size);
                wait(&status);
                printf("Parent Process Completed ...\n");
        }
        return 0;
            }

int split ( int a[ ], int lower, int upper )
{
        int i, p, q, t ;
        p = lower + 1 ;
        q = upper ;
        i = a[lower] ;
        while ( q >= p )
        {
                while ( a[p] < i )
                        p++ ;
                while ( a[q] > i )
                        q-- ;
                if ( q > p )
                {
                        t = a[p] ;
                        a[p] = a[q] ;
                        a[q] = t ;
                }
        }
        t = a[lower] ;
        a[lower] = a[q] ;
        a[q] = t ;
        return q ;
}
void quickSort(int a[],int lower, int upper)
     int i ;
     if ( upper > lower )
     {
             i = split ( a, lower, upper ) ;
             quickSort ( a, lower, i - 1 ) ;
             quickSort ( a, i + 1, upper ) ;
     }
}
-- INSERT --                                                                                      153,2         Bot
```



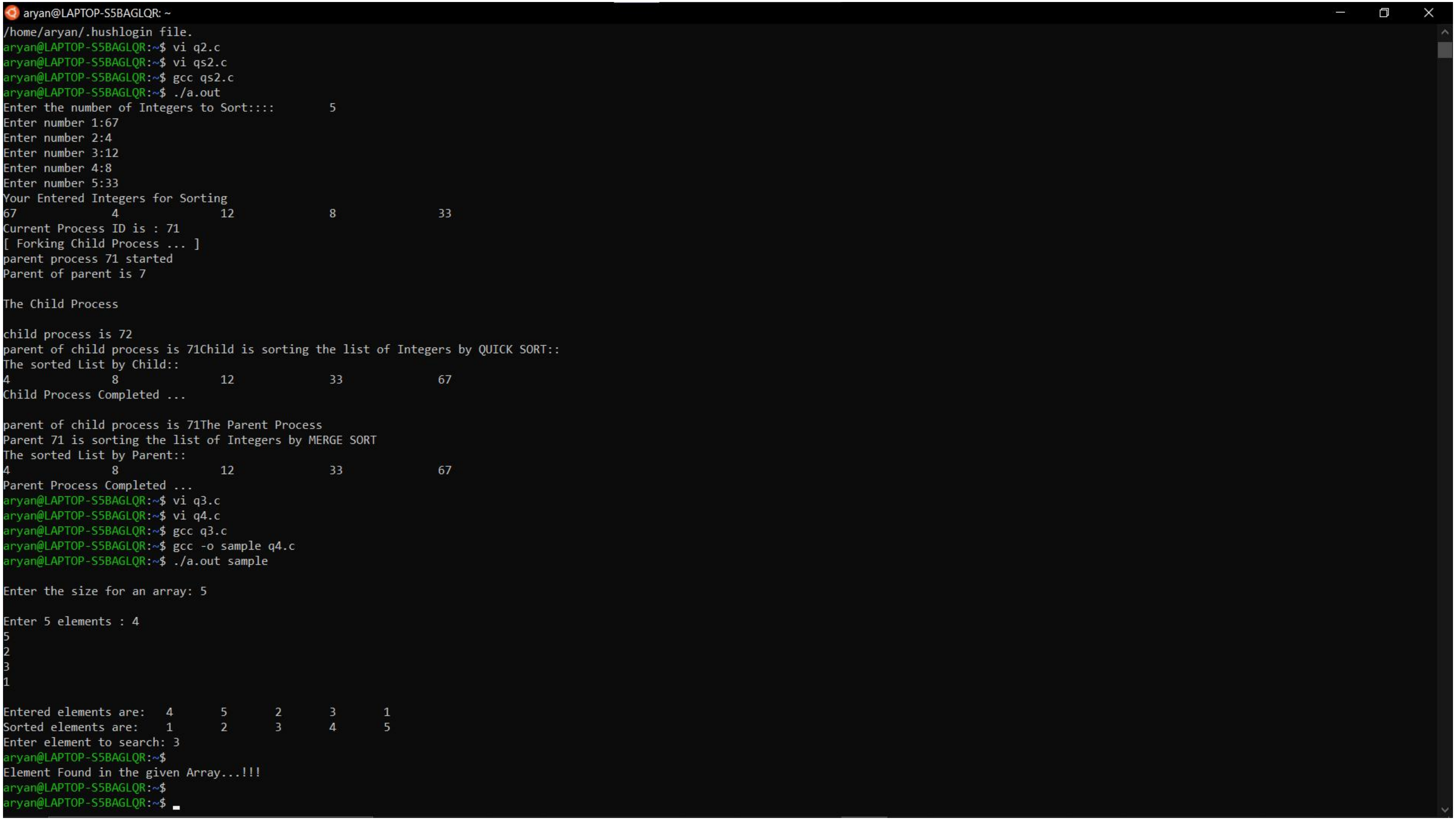
```
  System information as of Wed Mar 24 00:45:41 IST 2021

  System load:            0.52
  Usage of /home:         unknown
  Memory usage:           31%
  Swap usage:             0%
  Processes:              7
  Users logged in:        0
  IPv4 address for wifi0: 192.168.29.39
  IPv6 address for wifi0: 2405:201:7000:9909:7198:7c20:4113:6163
  IPv6 address for wifi0: 2405:201:7000:9909:a864:9f5d:83b:df30

29 updates can be installed immediately.
17 of these updates are security updates.
To see these additional updates run: apt list --upgradable



This message is shown once a day. To disable it please create the
/home/aryan/.hushlogin file.
aryan@LAPTOP-S5BAGLQR:~$ vi q2.c
aryan@LAPTOP-S5BAGLQR:~$ vi qs2.c
aryan@LAPTOP-S5BAGLQR:~$ gcc qs2.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out
Enter the number of Integers to Sort::::        5
Enter number 1:67
Enter number 2:4
Enter number 3:12
Enter number 4:8
Enter number 5:33
Your Entered Integers for Sorting
67              4               12              8               33
Current Process ID is : 71
[ Forking Child Process ... ]
parent process 71 started
Parent of parent is 7

The Child Process

child process is 72
parent of child process is 71Child is sorting the list of Integers by QUICK SORT::
The sorted List by Child::
4               8               12              33              67
Child Process Completed ...

parent of child process is 71The Parent Process
Parent 71 is sorting the list of Integers by MERGE SORT
The sorted List by Parent::
4               8               12              33              67
Parent Process Completed ...
aryan@LAPTOP-S5BAGLQR:~$
```



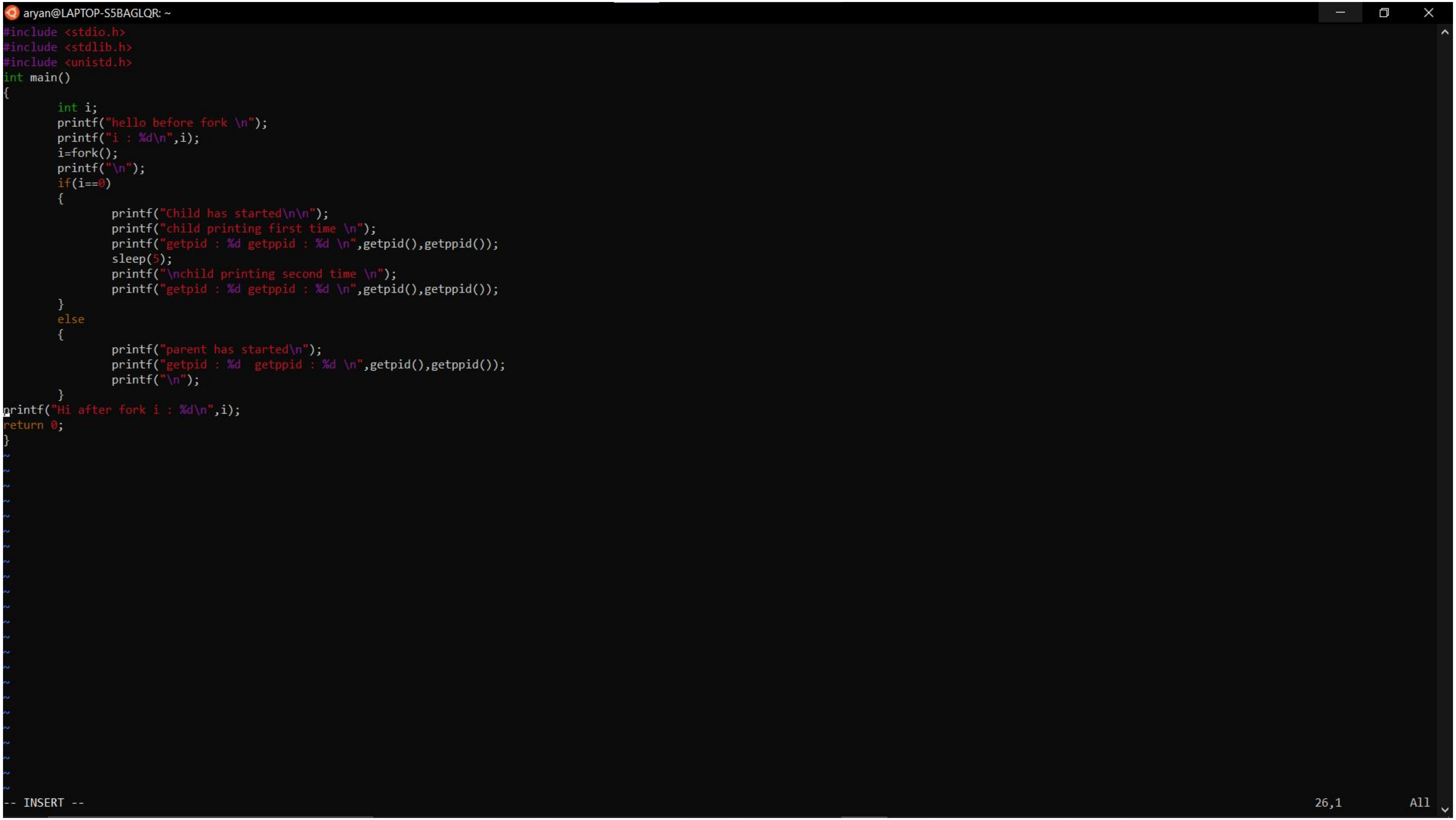
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
int main(int argc, char *argv[])
{
int val[10],ele;
pid_t pid;
char* cval[10];
char *newenviron[] = { NULL };
int i,j,n,temp;
printf("\nEnter the size for an array: ");
scanf("%d",&n);
printf("\nEnter %d elements : ", n);
for(i=0;i<n;i++)
        scanf("%d",&val[i]);
printf("\nEntered elements are: ");
for(i=0;i<n;i++)
        printf("\t%d",val[i]);
for(i=1;i<n;i++)
{
        for(j=0;j<n-1;j++)
        {
                if(val[j]>val[j+1])
                {
                        temp=val[j];
                        val[j]=val[j+1];
                        val[j+1]=temp;
                }
        }
}
printf("\nSorted elements are: ");
for(i=0;i<n;i++)
printf("\t%d",val[i]);
printf("\nEnter element to search: ");
scanf("%d",&ele);
val[i] = ele;
for (i=0; i < n+1; i++)
{
        char a[sizeof(int)];
        snprintf(a, sizeof(int), "%d", val[i]);
        cval[i] = malloc(sizeof(a));
        strcpy(cval[i], a);
}
cval[i]=NULL;
pid=fork();
if(pid==0)
{
        execve(argv[1], cval, newenviron);
        perror("Error in execve call...");
}
-- INSERT --                                                                                                   1,1           Top
```



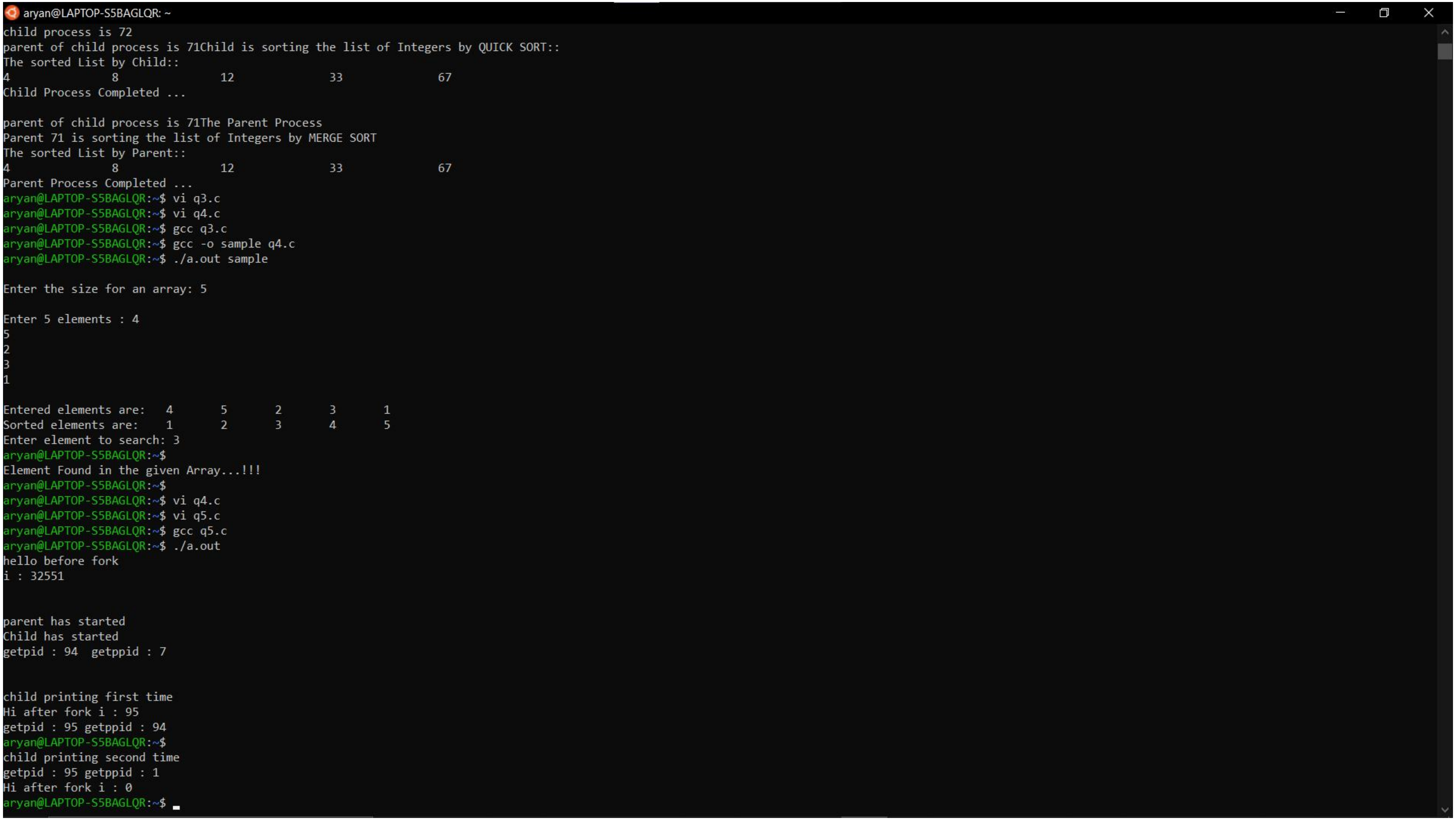
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char *argv[],char *en[])
{
        int i,j,c,ele;
        int arr[argc];
        for (j = 0; j < argc-1; j++)
        {
                int n=atoi(argv[j]);
                arr[j]=n;
        }
        ele=atoi(argv[j]);
        i=0;
        j=argc-1;
        c=(i+j)/2;
        while(arr[c]!=ele && i<=j)
        {
                if(ele > arr[c])
                        i = c+1;
                else
                        j = c-1;
                c = (i+j)/2;
        }
        if(i<=j)
                printf("\nElement Found in the given Array...!!!\n");
        else
                printf("\nElement Not Found in the given Array...!!!\n");
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                                                                     29,1          All
```



```
/home/aryan/.hushlogin file.
aryan@LAPTOP-S5BAGLQR:~$ vi q2.c
aryan@LAPTOP-S5BAGLQR:~$ vi qs2.c
aryan@LAPTOP-S5BAGLQR:~$ gcc qs2.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out
Enter the number of Integers to Sort::::        5
Enter number 1:67
Enter number 2:4
Enter number 3:12
Enter number 4:8
Enter number 5:33
Your Entered Integers for Sorting
67              4               12              8               33
Current Process ID is : 71
[ Forking Child Process ... ]
parent process 71 started
Parent of parent is 7

The Child Process

child process is 72
parent of child process is 71Child is sorting the list of Integers by QUICK SORT::
The sorted List by Child::
4               8               12              33              67
Child Process Completed ...

parent of child process is 71The Parent Process
Parent 71 is sorting the list of Integers by MERGE SORT
The sorted List by Parent::
4               8               12              33              67
Parent Process Completed ...
aryan@LAPTOP-S5BAGLQR:~$ vi q3.c
aryan@LAPTOP-S5BAGLQR:~$ vi q4.c
aryan@LAPTOP-S5BAGLQR:~$ gcc q3.c
aryan@LAPTOP-S5BAGLQR:~$ gcc -o sample q4.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out sample

Enter the size for an array: 5

Enter 5 elements : 4
5
2
3
1

Entered elements are:   4       5       2       3       1
Sorted elements are:    1       2       3       4       5
Enter element to search: 3
aryan@LAPTOP-S5BAGLQR:~$
Element Found in the given Array...!!!
aryan@LAPTOP-S5BAGLQR:~$
aryan@LAPTOP-S5BAGLQR:~$
```



```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main()
{
        int i;
        printf("hello before fork \n");
        printf("i : %d\n",i);
        i=fork();
        printf("\n");
        if(i==0)
        {
                printf("Child has started\n\n");
                printf("child printing first time \n");
                printf("getpid : %d getppid : %d \n",getpid(),getppid());
                sleep(5);
                printf("\nchild printing second time \n");
                printf("getpid : %d getppid : %d \n",getpid(),getppid());
        }
        else
        {
                printf("parent has started\n");
                printf("getpid : %d  getppid : %d \n",getpid(),getppid());
                printf("\n");
        }
printf("Hi after fork i : %d\n",i);
return 0;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                                                        26,1          All
```



```
child process is 72
parent of child process is 71Child is sorting the list of Integers by QUICK SORT::
The sorted List by Child::
4               8               12              33              67
Child Process Completed ...

parent of child process is 71The Parent Process
Parent 71 is sorting the list of Integers by MERGE SORT
The sorted List by Parent::
4               8               12              33              67
Parent Process Completed ...
aryan@LAPTOP-S5BAGLQR:~$ vi q3.c
aryan@LAPTOP-S5BAGLQR:~$ vi q4.c
aryan@LAPTOP-S5BAGLQR:~$ gcc q3.c
aryan@LAPTOP-S5BAGLQR:~$ gcc -o sample q4.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out sample

Enter the size for an array: 5

Enter 5 elements : 4
5
2
3
1

Entered elements are:   4       5       2       3       1
Sorted elements are:    1       2       3       4       5
Enter element to search: 3
aryan@LAPTOP-S5BAGLQR:~$
Element Found in the given Array...!!!
aryan@LAPTOP-S5BAGLQR:~$
aryan@LAPTOP-S5BAGLQR:~$ vi q4.c
aryan@LAPTOP-S5BAGLQR:~$ vi q5.c
aryan@LAPTOP-S5BAGLQR:~$ gcc q5.c
aryan@LAPTOP-S5BAGLQR:~$ ./a.out
hello before fork
i : 32551


parent has started
Child has started
getpid : 94  getppid : 7


child printing first time
Hi after fork i : 95
getpid : 95 getppid : 94
aryan@LAPTOP-S5BAGLQR:~$
child printing second time
getpid : 95 getppid : 1
Hi after fork i : 0
aryan@LAPTOP-S5BAGLQR:~$
```