Mini Project

Somesh Pandey

November 16, 2022

Contents

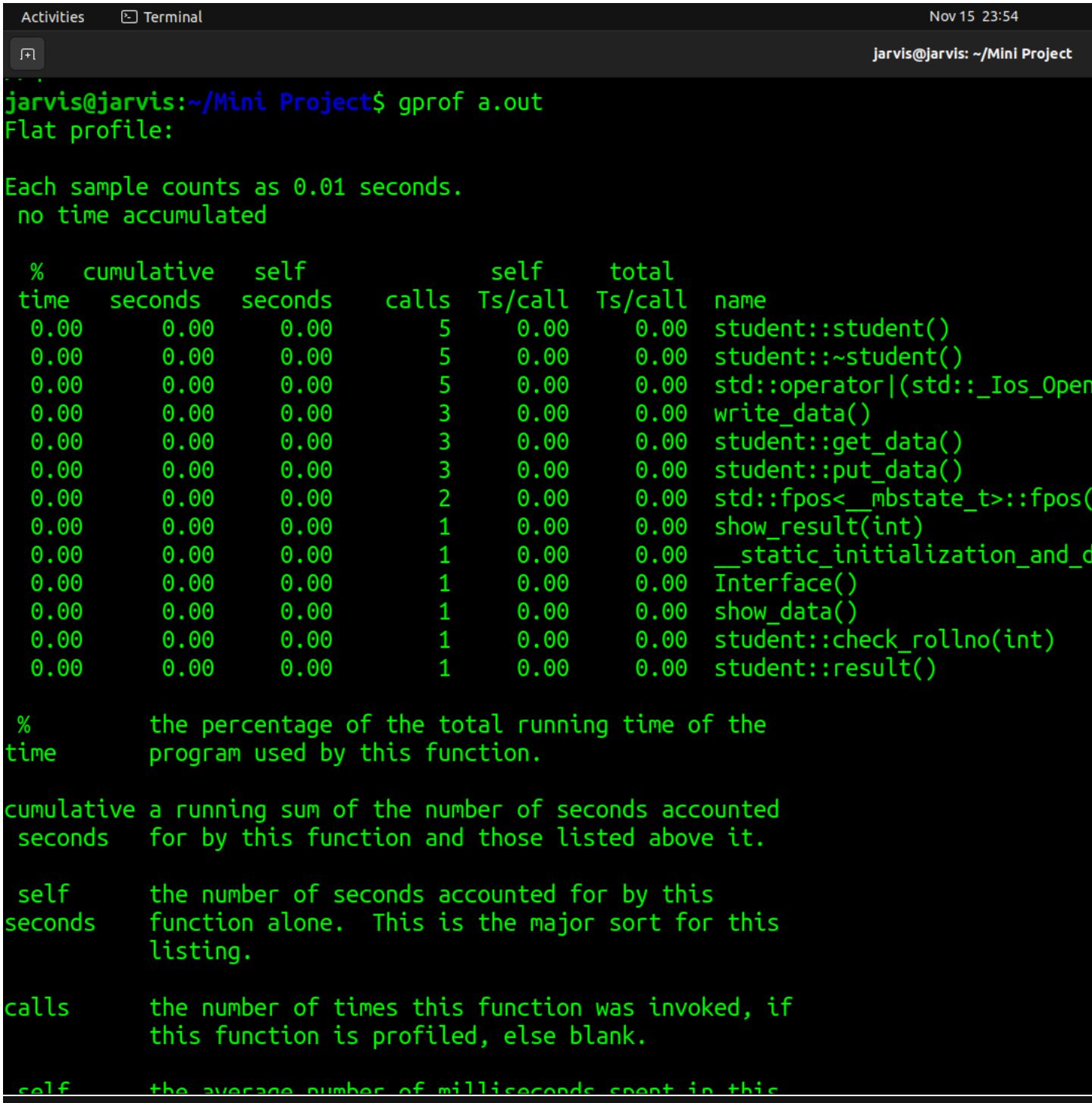
1 Objective of project

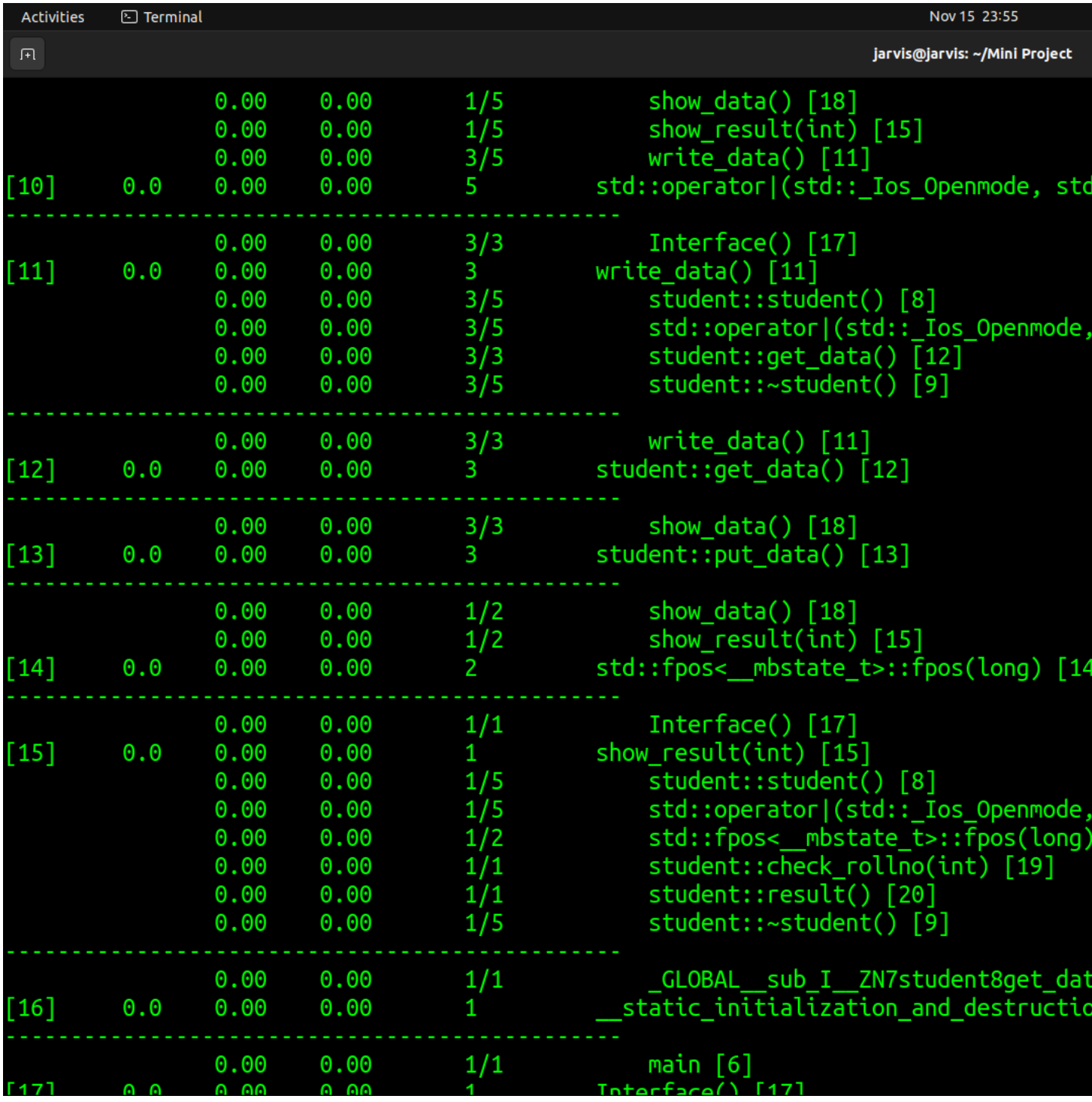
This project is based on the concept of Student Report System if which we can access and store data of student. This project is coded in C++ programming language and concept of Data Handling in C++ and Object oriented programming in C++.
File handling in C++ is a mechanism to store the output of a program in a file and help perform various operations on it. Files help store these data permanently on a storage device. The term "Data" is commonly referred to as known facts or information.
OOP stands for Object-Oriented Programming. Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.
This program takes name, roll number and marks in various subjects as input and prints the ouput like result and scorecard.

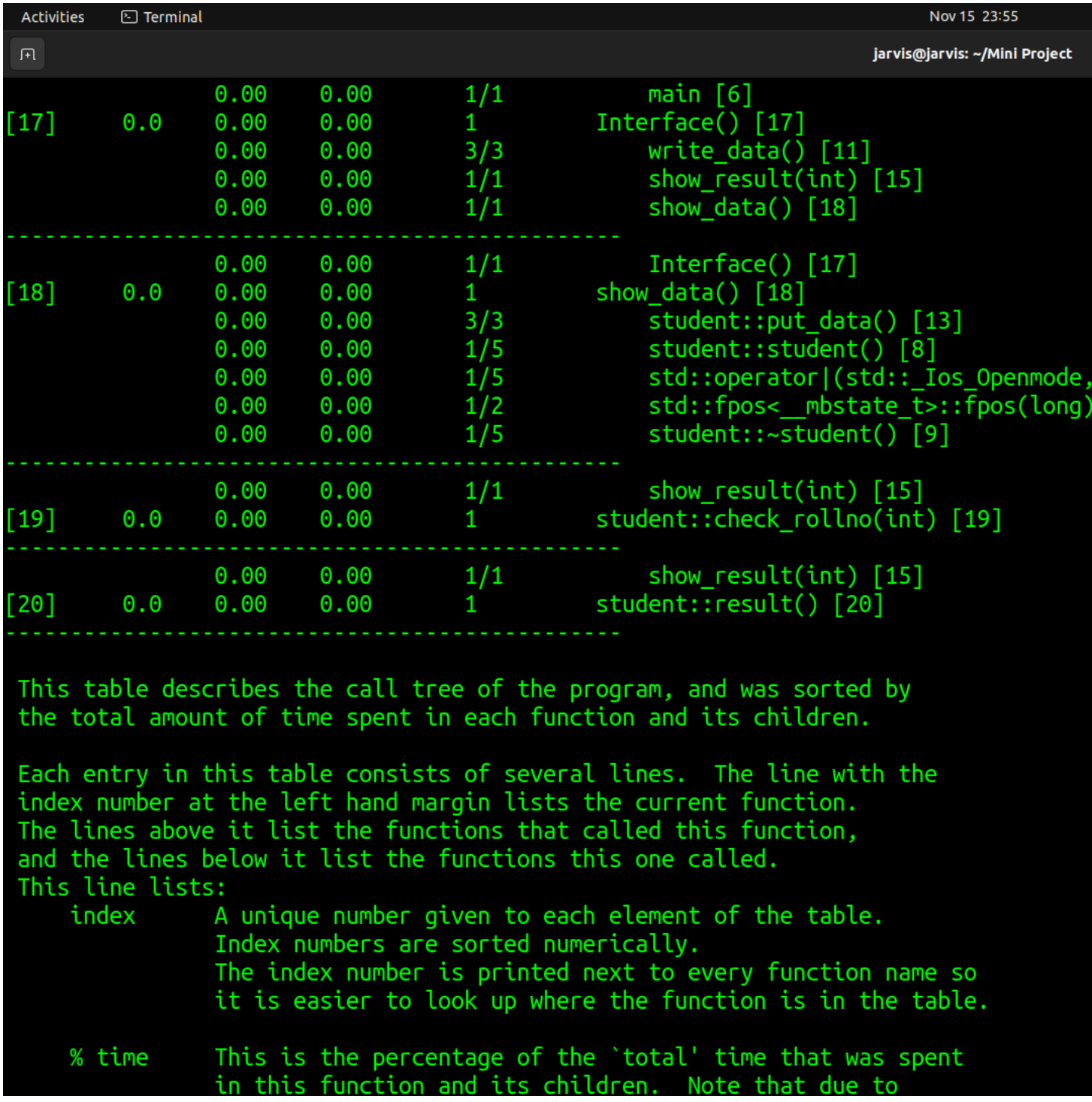
2 Function Description

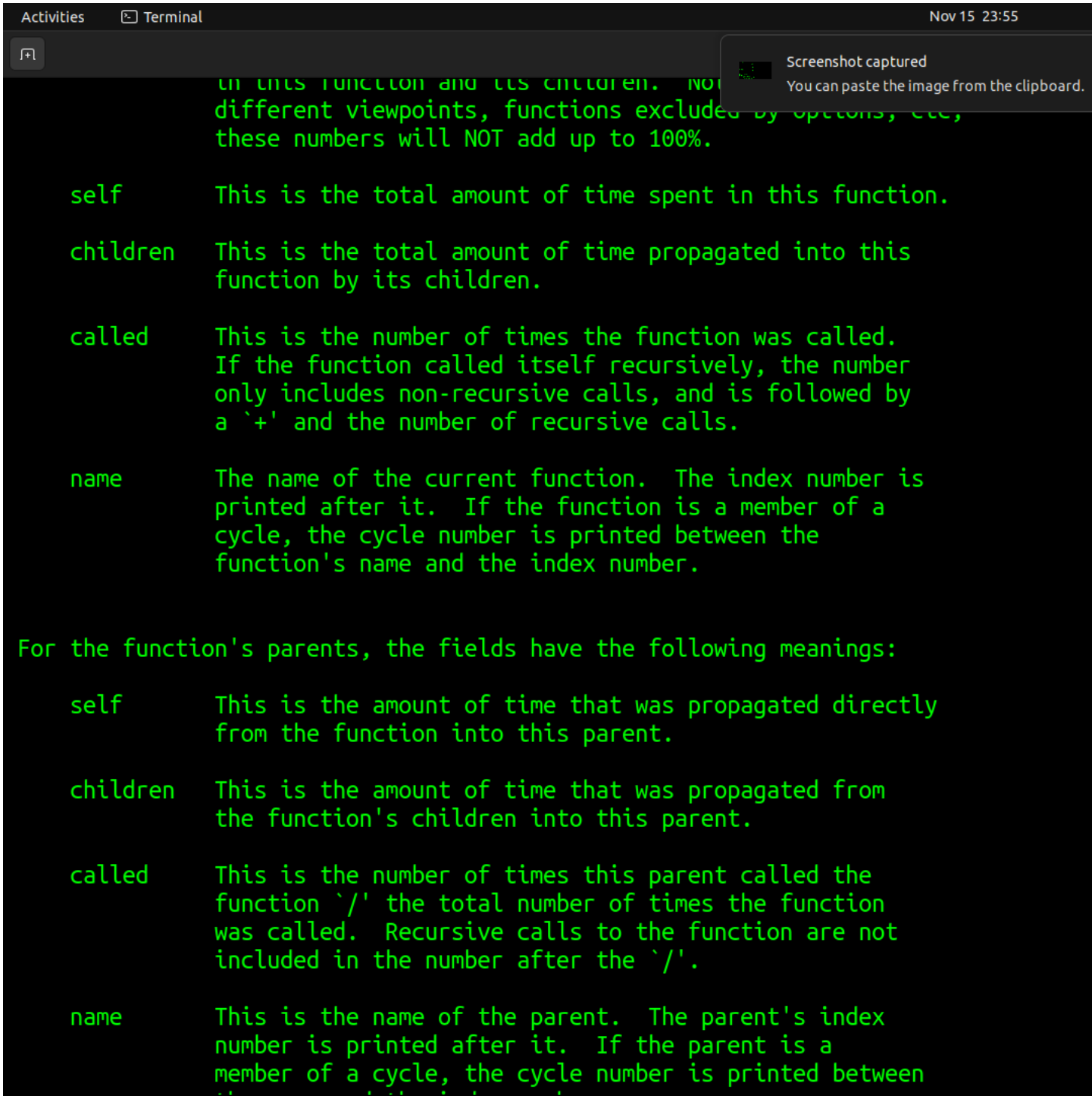
The first function is *getdata*() which is a class function which takes the input from the user.
The next function is *putdata*() which is also an class function which outputs the data of the user.
The next function is *result*() which is also an class function which prints the marks and percentage of the user. The next function is *checkRollNO*() which checks the given roll number is present in the user input.
The function *writedata*() is used to take data from user and open file name Student Report and put data in it.
The function *showdata*() is used to take data from file Student Report and print the output.
The function *showresult*() is used to open file Student Report and prints all its contents.
The function $Interface()$ shows the screen with various options to do in the file.
The *intmain*() calls the Interface.

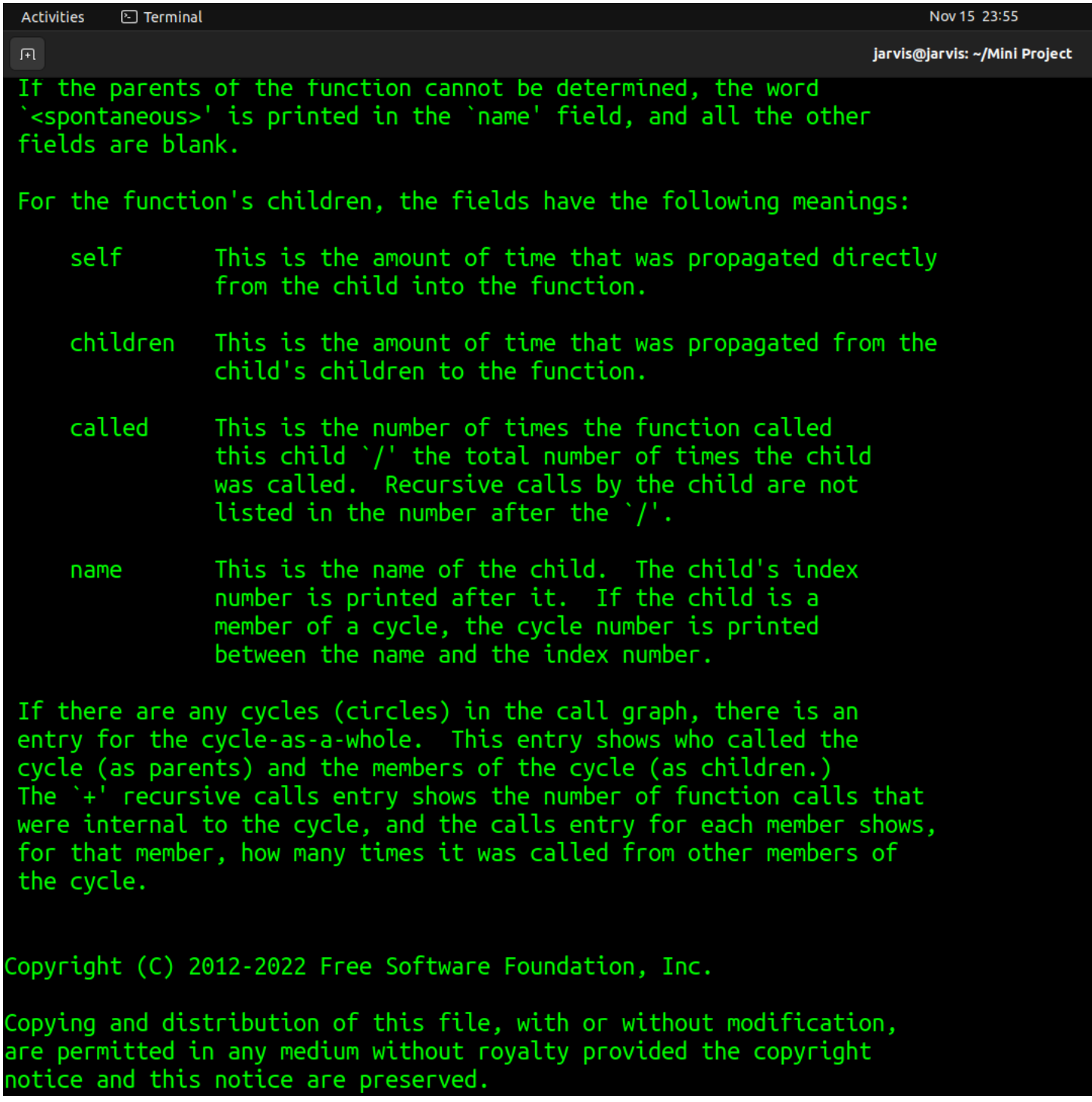
3 Profiling

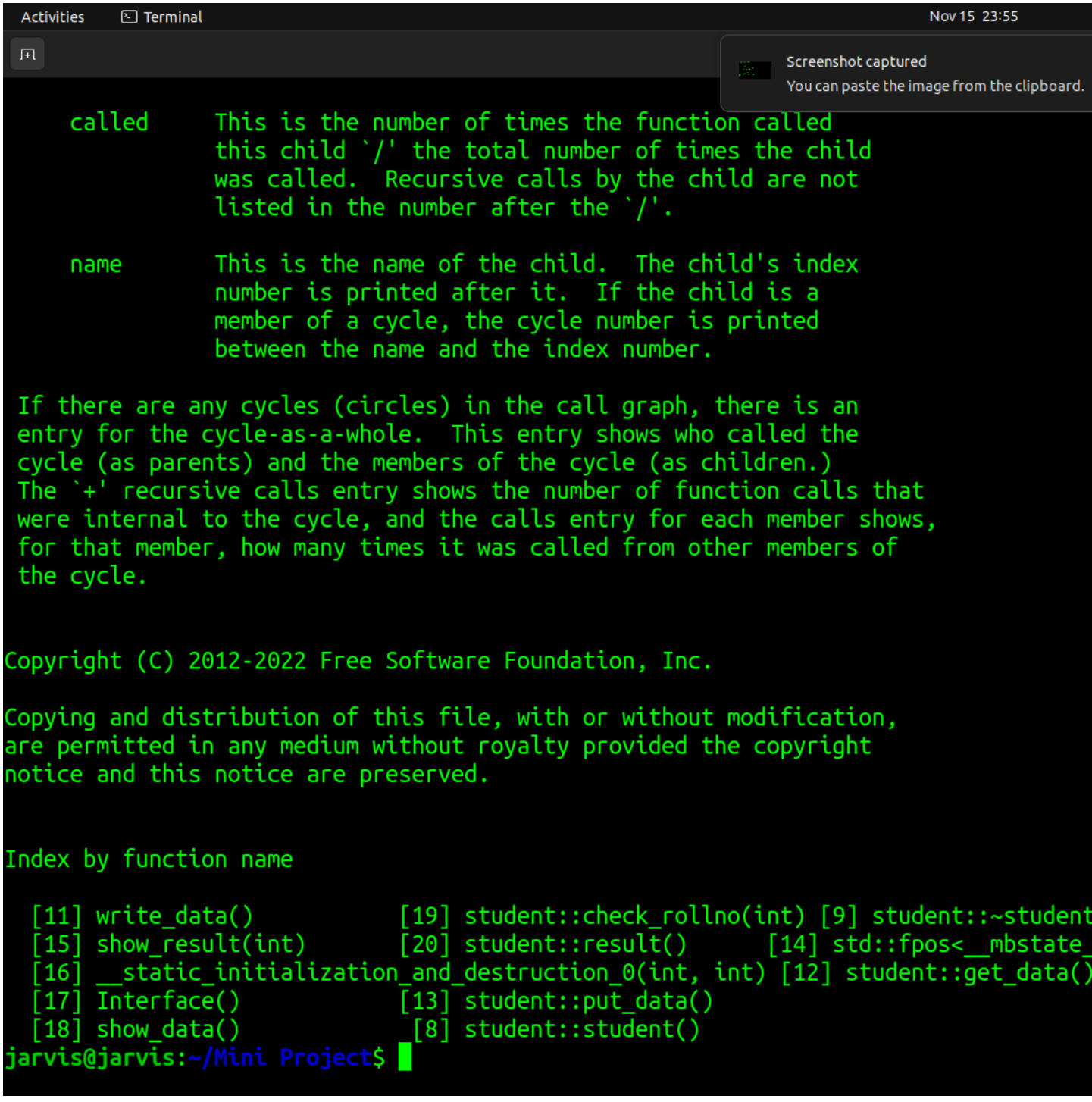

Activities
Terminal
Nov 15 23:54
jarvis@jarvis: ~/Mini Project
jarvis@jarvis:~/Mini Project$ gprof a.out
Flat profile:
Each sample counts as 0.01 seconds.
no time accumulated
% cumulative self self total
time seconds seconds calls Ts/call Ts/call name
0.00 0.00 0.00 5 0.00 0.00 student::student()
0.00 0.00 0.00 5 0.00 0.00 student::~student()
0.00 0.00 0.00 5 0.00 0.00 std::operator|(std::_Ios_Open
0.00 0.00 0.00 3 0.00 0.00 write_data()
0.00 0.00 0.00 3 0.00 0.00 student::get_data()
0.00 0.00 0.00 3 0.00 0.00 student::put_data()
0.00 0.00 0.00 2 0.00 0.00 std::fpos<__mbstate_t>::fpos(
0.00 0.00 0.00 1 0.00 0.00 show_result(int)
0.00 0.00 0.00 1 0.00 0.00 __static_initialization_and_d
0.00 0.00 0.00 1 0.00 0.00 Interface()
0.00 0.00 0.00 1 0.00 0.00 show_data()
0.00 0.00 0.00 1 0.00 0.00 student::check_rollno(int)
0.00 0.00 0.00 1 0.00 0.00 student::result()
% the percentage of the total running time of the
time program used by this function.
cumulative a running sum of the number of seconds accounted
seconds for by this function and those listed above it.
self the number of seconds accounted for by this
seconds function alone. This is the major sort for this
listing.
calls the number of times this function was invoked, if
this function is profiled, else blank.
Activities
Terminal
Nov 15 23:55
jarvis@jarvis: ~/Mini Project
else blank.
total the average number of milliseconds spent in this
ms/call function and its descendents per call, if this
function is profiled, else blank.
name the name of the function. This is the minor sort
for this listing. The index shows the location of
the function in the gprof listing. If the index is
in parenthesis it shows where it would appear in
the gprof listing if it were to be printed.


Activities
Terminal
Nov 15 23:55
jarvis@jarvis: ~/Mini Project
0.00 0.00 1/5 show_data() [18]
0.00 0.00 1/5 show_result(int) [15]
0.00 0.00 3/5 write_data() [11]
[10] 0.0 0.00 0.00 5 std::operator|(std::_Ios_Openmode, std
0.00 0.00 3/3 Interface() [17]
[11] 0.0 0.00 0.00 3 write_data() [11]
0.00 0.00 3/5 student::student() [8]
0.00 0.00 3/5 std::operator|(std::_Ios_Openmode,
0.00 0.00 3/3 student::get_data() [12]
0.00 0.00 3/5 student::~student() [9]
0.00 0.00 3/3 write_data() [11]
[12] 0.0 0.00 0.00 3 student::get_data() [12]
0.00 0.00 3/3 show_data() [18]
[13] 0.0 0.00 0.00 3 student::put_data() [13]
0.00 0.00 1/2 show_data() [18]
0.00 0.00 1/2 show_result(int) [15]
[14] 0.0 0.00 0.00 2 std::fpos<__mbstate_t>::fpos(long) [14
0.00 0.00 1/1 Interface() [17]
[15] 0.0 0.00 0.00 1 show_result(int) [15]
0.00 0.00 1/5 student::student() [8]
0.00 0.00 1/5 std::operator|(std::_Ios_Openmode,
0.00 0.00 1/2 std::fpos<__mbstate_t>::fpos(long)
0.00 0.00 1/1 student::check_rollno(int) [19]
0.00 0.00 1/1 student::result() [20]
0.00 0.00 1/5 student::~student() [9]
0.00 0.00 1/1 _GLOBAL__sub_I__ZN7student8get_dat
[16] 0.0 0.00 0.00 1 __static_initialization_and_destructio
0.00 0.00 1/1 main [6]

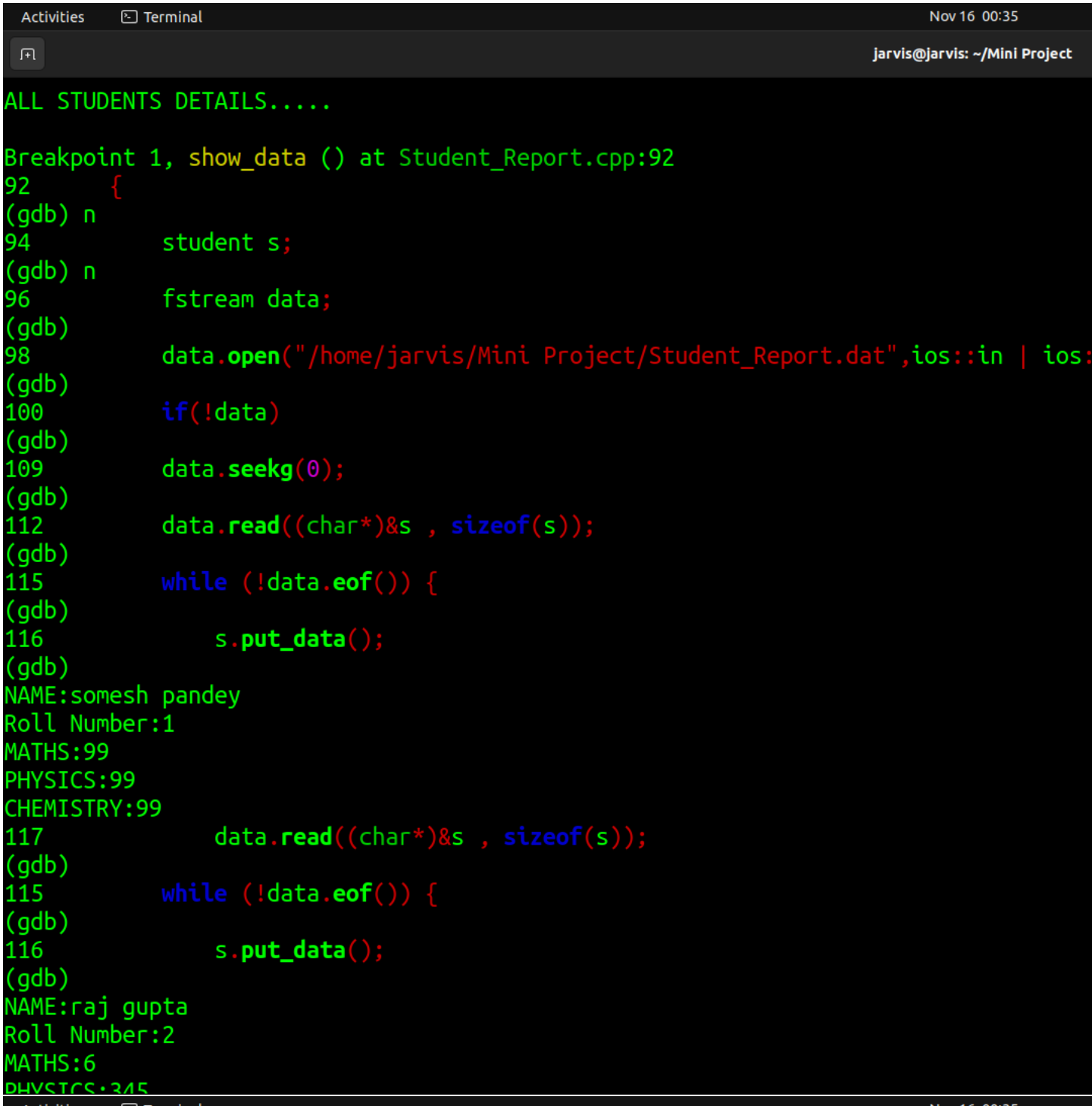

Activities
Terminal
Nov 15 23:55
jarvis@jarvis: ~/Mini Project
0.00 0.00 1/1 main [6]
[17] 0.0 0.00 0.00 1 Interface() [17]
0.00 0.00 3/3 write_data() [11]
0.00 0.00 1/1 show_result(int) [15]
0.00 0.00 1/1 show_data() [18]
0.00 0.00 1/1 Interface() [17]
[18] 0.0 0.00 0.00 1 show_data() [18]
0.00 0.00 3/3 student::put_data() [13]
0.00 0.00 1/5 student::student() [8]
0.00 0.00 1/5 std::operator|(std::_Ios_Openmode,
0.00 0.00 1/2 std::fpos<__mbstate_t>::fpos(long)
0.00 0.00 1/5 student::~student() [9]
0.00 0.00 1/1 show_result(int) [15]
[19] 0.0 0.00 0.00 1 student::check_rollno(int) [19]
0.00 0.00 1/1 show_result(int) [15]
[20] 0.0 0.00 0.00 1 student::result() [20]
This table describes the call tree of the program, and was sorted by
the total amount of time spent in each function and its children.
Each entry in this table consists of several lines. The line with the
index number at the left hand margin lists the current function.
The lines above it list the functions that called this function,
and the lines below it list the functions this one called.
This line lists:
index A unique number given to each element of the table.
Index numbers are sorted numerically.
The index number is printed next to every function name so
it is easier to look up where the function is in the table.
% time This is the percentage of the `total' time that was spent
in this function and its children. Note that due to

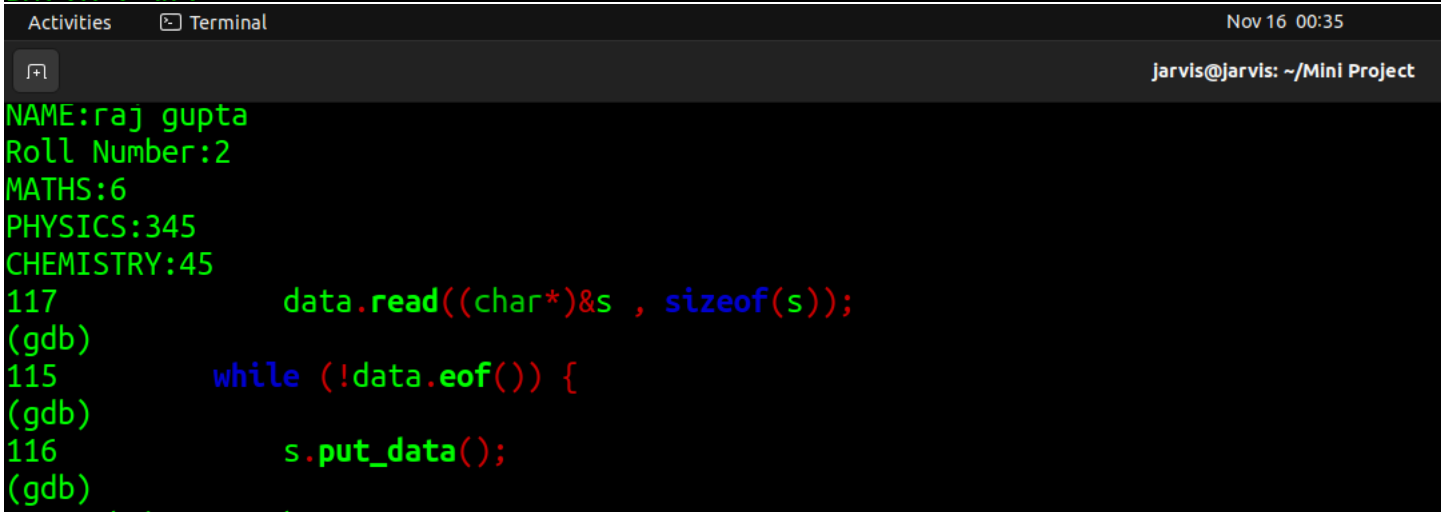

Activities
Terminal
Nov 15 23:55
Screenshot captured
You can paste the image from the clipboard.
in this function and its children. Not
different viewpoints, functions excluded by options, etc,
these numbers will NOT add up to 100%.
self This is the total amount of time spent in this function.
children This is the total amount of time propagated into this function by its children.
called This is the number of times the function was called. If the function called itself recursively, the number only includes non-recursive calls, and is followed by a `+' and the number of recursive calls.
name The name of the current function. The index number is printed after it. If the function is a member of a cycle, the cycle number is printed between the function's name and the index number.
For the function's parents, the fields have the following meanings:
self This is the amount of time that was propagated directly from the function into this parent.
children This is the amount of time that was propagated from the function's children into this parent.
called This is the number of times this parent called the function `/' the total number of times the function was called. Recursive calls to the function are not included in the number after the `/'.
name This is the name of the parent. The parent's index number is printed after it. If the parent is a member of a cycle, the cycle number is printed between

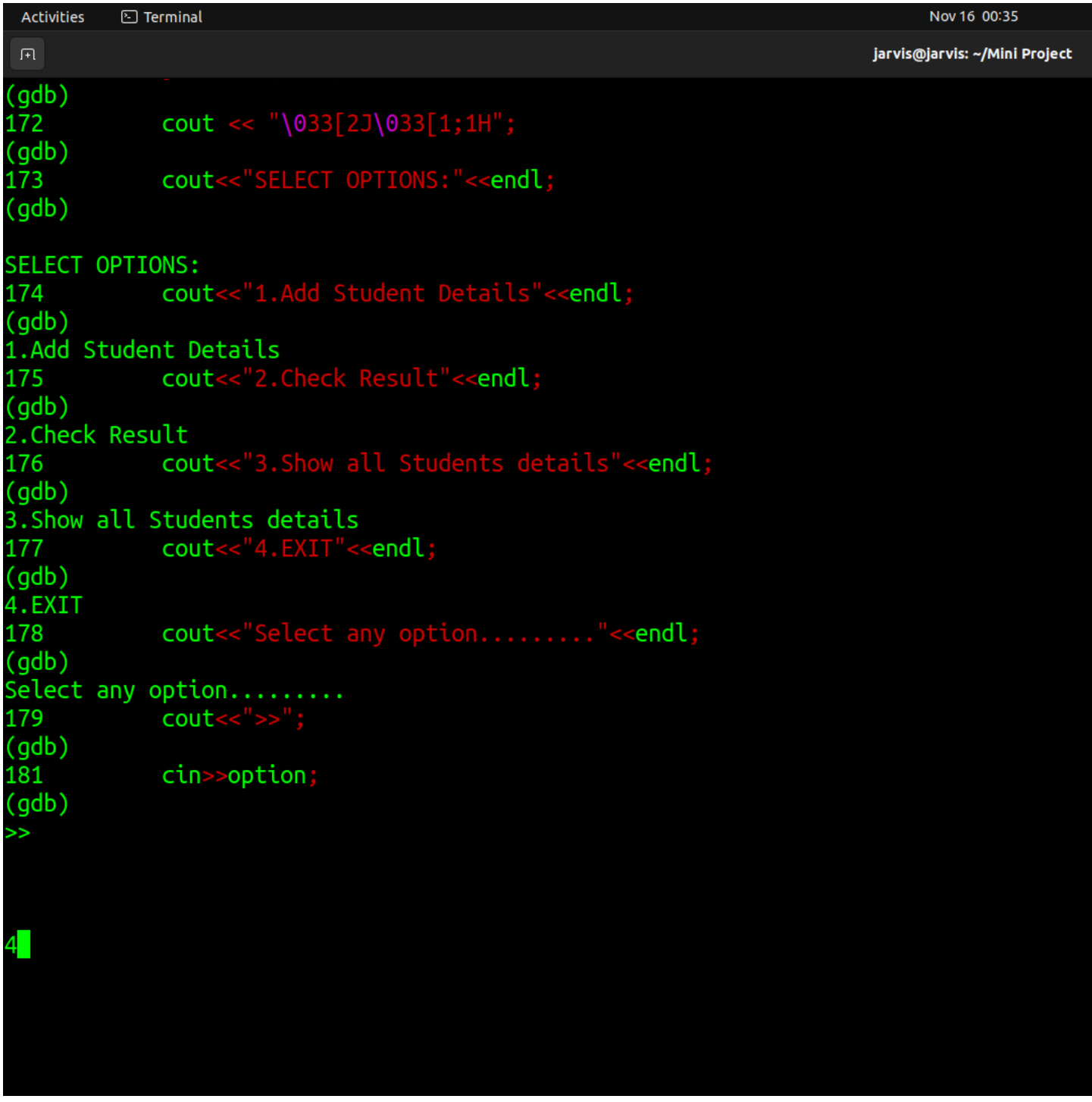

Activities
Terminal
Nov 15 23:55
jarvis@jarvis: ~/Mini Project
If the parents of the function cannot be determined, the word
`<spontaneous>' is printed in the `name' field, and all the other
fields are blank.
For the function's children, the fields have the following meanings:
self This is the amount of time that was propagated directly
from the child into the function.
children This is the amount of time that was propagated from the
child's children to the function.
called This is the number of times the function called
this child `/' the total number of times the child
was called. Recursive calls by the child are not
listed in the number after the `/'.
name This is the name of the child. The child's index
number is printed after it. If the child is a
member of a cycle, the cycle number is printed
between the name and the index number.
If there are any cycles (circles) in the call graph, there is an
entry for the cycle-as-a-whole. This entry shows who called the
cycle (as parents) and the members of the cycle (as children.)
The `+' recursive calls entry shows the number of function calls that
were internal to the cycle, and the calls entry for each member shows,
for that member, how many times it was called from other members of
the cycle.
Copyright (C) 2012-2022 Free Software Foundation, Inc.
Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.

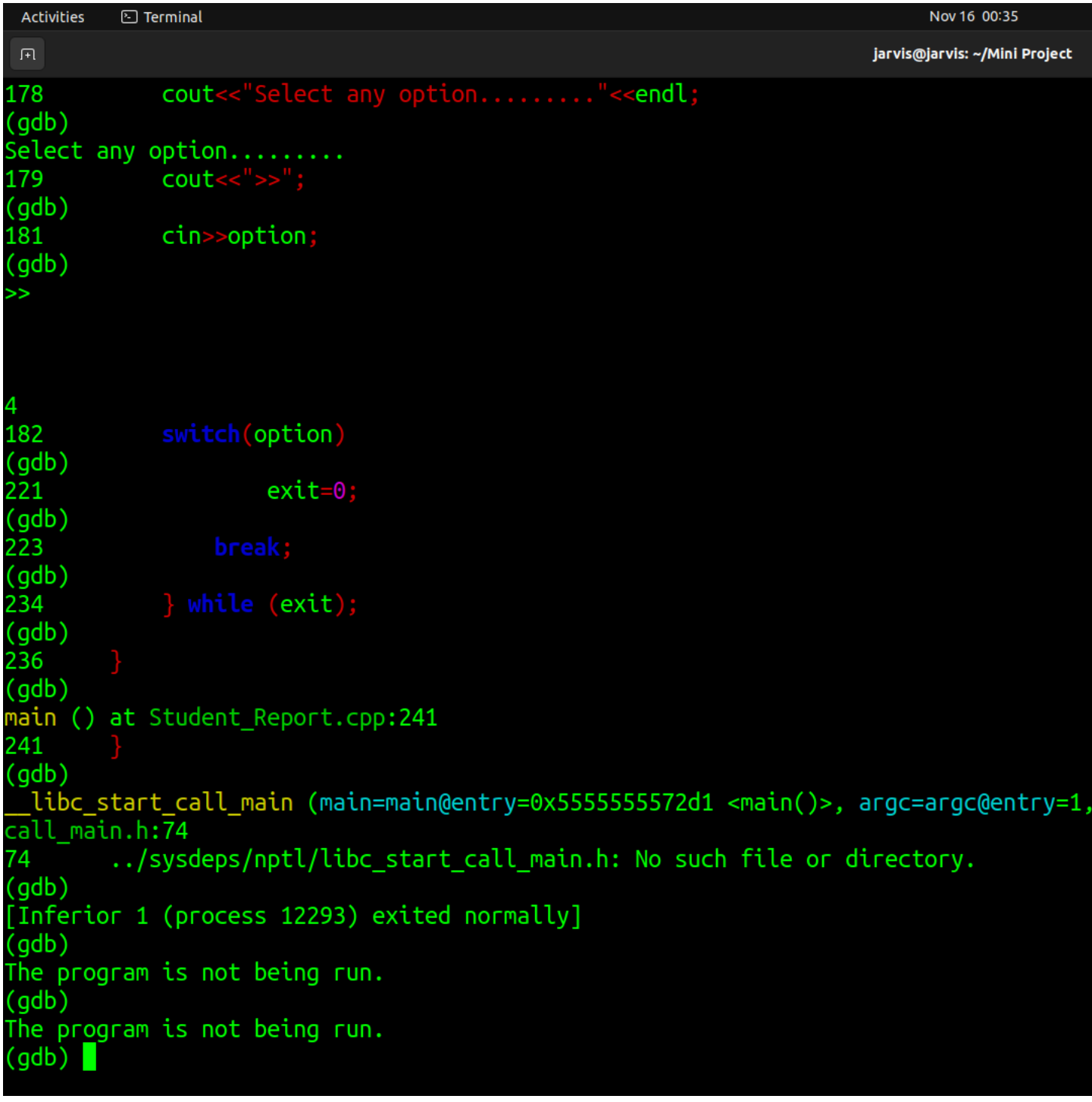

Activities
Terminal
Nov 15 23:55
Screenshot captured
You can paste the image from the clipboard.
called This is the number of times the function called
this child `/' the total number of times the child
was called. Recursive calls by the child are not
listed in the number after the `/'.
name This is the name of the child. The child's index
number is printed after it. If the child is a
member of a cycle, the cycle number is printed
between the name and the index number.
If there are any cycles (circles) in the call graph, there is an
entry for the cycle-as-a-whole. This entry shows who called the
cycle (as parents) and the members of the cycle (as children.)
The `+' recursive calls entry shows the number of function calls that
were internal to the cycle, and the calls entry for each member shows,
for that member, how many times it was called from other members of
the cycle.
Copyright (C) 2012-2022 Free Software Foundation, Inc.
Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.
Index by function name
[11] write_data() [19] student::check_rollno(int) [9] student::~student
[15] show_result(int) [20] student::result() [14] std::fpos<__mbstate_
[16] __static_initialization_and_destruction_0(int, int) [12] student::get_data()
[17] Interface() [13] student::put_data()
[18] show_data() [8] student::student()
jarvis@jarvis:~/Mini Project$

4 Debugging


Activities
Terminal
Nov 16 00:35
jarvis@jarvis: ~/Mini Project
ALL STUDENTS DETAILS.....
Breakpoint 1, show_data () at Student_Report.cpp:92
92 {
(gdb) n
94 student s;
(gdb) n
96 fstream data;
(gdb)
98 data.open("/home/jarvis/Mini Project/Student_Report.dat",ios::in | ios:
(gdb)
100 if(!data)
(gdb)
109 data.seekg(0);
(gdb)
112 data.read((char*)&s , sizeof(s));
(gdb)
115 while (!data.eof()) {
(gdb)
116 s.put_data();
(gdb)
NAME:somesh pandey
Roll Number:1
MATHS:99
PHYSICS:99
CHEMISTRY:99
117 data.read((char*)&s , sizeof(s));
(gdb)
115 while (!data.eof()) {
(gdb)
116 s.put_data();
(gdb)
NAME:raj gupta
Roll Number:2
MATHS:6
PHYSICS:345


Activities
Terminal
Nov 16 00:35
jarvis@jarvis: ~/Mini Project
NAME:raj gupta
Roll Number:2
MATHS:6
PHYSICS:345
CHEMISTRY:45
117 data.read((char*)&s , sizeof(s));
(gdb)
115 while (!data.eof()) {
(gdb)
116 s.put_data();
(gdb)


Activities
Terminal
Nov 16 00:35
jarvis@jarvis: ~/Mini Project
(gdb)
172 cout << "\033[2J\033[1;1H";
(gdb)
173 cout<<"SELECT OPTIONS:"<<endl;
(gdb)
SELECT OPTIONS:
174 cout<<"1.Add Student Details"<<endl;
(gdb)
1.Add Student Details
175 cout<<"2.Check Result"<<endl;
(gdb)
2.Check Result
176 cout<<"3.Show all Students details"<<endl;
(gdb)
3.Show all Students details
177 cout<<"4.EXIT"<<endl;
(gdb)
4.EXIT
178 cout<<"Select any option....."<<endl;
(gdb)
Select any option.....
179 cout<<">>";
(gdb)
181 cin>>option;
(gdb)
>>
4


Activities
Terminal
Nov 16 00:35
jarvis@jarvis: ~/Mini Project
178 cout<<"Select any option........."<<endl;
(gdb)
Select any option.........
179 cout<<">>";
(gdb)
181 cin>>option;
(gdb)
>>
4
182 switch(option)
(gdb)
221 exit=0;
(gdb)
223 break;
(gdb)
234 } while (exit);
(gdb)
236 }
(gdb)
main () at Student_Report.cpp:241
241 }
(gdb)
__libc_start_call_main (main=main@entry=0x5555555572d1 <main()>, argc=argc@entry=1,
call_main.h:74
74 ../sysdeps/nptl/libc_start_call_main.h: No such file or directory.
(gdb)
[Inferior 1 (process 12293) exited normally]
(gdb)
The program is not being run.
(gdb)
The program is not being run.
(gdb)

5 Code

Code:

```
#include<iostream>
#include<fstream>
using namespace std;



// class of student
class student{
    int roll_no; //roll number of student
    string name; // name of the student
    float marks_maths; // marks in maths
    float marks_physics; // marks in physics
    float marks_chemistry; // marks in physics
    float total_marks; // total marks out of 300
    float percent; // percentage
public:
    void get_data();
    void put_data();
    void result();
    bool check_rollno(int);
};

//class funtion to input details of the students
void student :: get_data()
{
    cout<<"Enter name:";
    cin.ignore();
    getline(cin, name);
    cout<<"Enter roll number:";
    cin>>roll_no;
    cout<<"Enter maths marks:";
    cin>>marks_maths;
    cout<<"Enter physics:";
    cin>>marks_physics;
    cout<<"Enter chemistry marks:";
    cin>>marks_chemistry;
}

void student :: put_data()
{
    cout<<"NAME:"<<name<<endl;
    cout<<"Roll Number:"<<roll_no<<endl;
    cout<<"MATHS:"<<marks_maths<<endl;
```

```
    cout<<"PHYSICS:"<<marks_physics<<endl;
    cout<<"CHEMISTRY:"<<marks_chemistry<<endl;
}

void student :: result()
{
    total_marks=marks_maths+marks_physics+marks_chemistry;
    percent=total_marks/3;
    cout<<"NAME:"<<name<<endl;
    cout<<"Roll Number:"<<roll_no<<endl;
    cout<<"MATHS:"<<marks_maths<<endl;
    cout<<"PHYSICS:"<<marks_physics<<endl;
    cout<<"CHEMISTRY:"<<marks_chemistry<<endl;
    cout<<"The pertage is:"<<percent;
}

//to check the roll number
bool student :: check_rollno(int num)
{
        if(num==roll_no)
        {
            return true;
        }
        else return false;
}

// function to input details to the file
void write_data()
{
    // student object
    student s;

    fstream data;
    // opening file
    data.open("/home/jarvis/Mini_Project/Student_Report.dat" , ios::app | ios:: bi

    //taking values of student
    s.get_data();

    //writing data to file
    data.write((char*)&s , sizeof(s));

    //closing files
    data.close();
}

void show_data()
{
    //student object
```

```
    student s;

    fstream data;
    // opening file
    data.open("/home/jarvis/Mini Project/Student Report.dat",ios::in | ios:: binary

    if(!data)
        {
                cout<<"File could not be open !! Press any Key...";
                cin.ignore();
                cin.get();
                return;
        }

    //pointer to start
    data.seekg(0);

    //reading data
    data.read((char*)&s , sizeof(s));

    // all the things in file
    while (!data.eof()) {
        s.put_data();
        data.read((char*)&s , sizeof(s));
    }

    //close file
    data.close();
}

void show_result(int roll_no)
{
    //student object
    student s;

    fstream data;
    // opening file
    data.open("/home/jarvis/Mini Project/Student Report.dat",ios::in | ios::binary

    if(!data)
        {
                cout<<"File could not be open !! Press any Key...";
                cin.ignore();
                cin.get();
                return;
        }

    //pointer to start
    data.seekg(0);
```

```
    //reading data
    data.read((char*)&s , sizeof(s));

    while(!data.eof())
    {
        if(s.check_rollno(roll_no))
        {
            s.result();
            break;
        }
        data.read((char*)&s , sizeof(s));
    }
}

// This is the interface of the program
void Interface()
{
    cout << "\033[2J\033[1;1H";
    cout<<endl;
    cout<<"**********************************************"<<endl;
    cout<<"*            Student Report System           *"<<endl;
    cout<<"**********************************************"<<endl;
    cout<<endl<<endl;
    cout<<"      Please enter any key to continue ....."<<endl;
    cin.ignore();
    int exit=1;
    do
    {
    cout << "\033[2J\033[1;1H";
    cout<<"SELECT OPTIONS:"<<endl;
    cout<<"1.Add Student Details"<<endl;
    cout<<"2.Check Result"<<endl;
    cout<<"3.Show all Students details"<<endl;
    cout<<"4.EXIT"<<endl;
    cout<<"Select any option ........."<<endl;
    cout<<">>";
    int option;
    cin>>option;
    switch(option)
    {
        case 1:
        {
            cout << "\033[2J\033[1;1H";
            cout<<"Add Student Details"<<endl;
            write_data();
            cout<<"DATA added successfully"<<endl;
            cin.ignore();
                    cin.get();
```

```
        }
        break;

        case 2:
        {
            cout << "\033[2J\033[1;1H";
            cout<<"Check Result"<<endl;
            cout<<"Enter Roll Number:"<<endl;
            int r_no;
            cin>>r_no;
            show_result(r_no);
            cin.ignore();
                    cin.get();
        }
        break;

        case 3:
        {
            cout << "\033[2J\033[1;1H";
            cout<<"ALL STUDENTS DETAILS....."<<endl;
            show_data();
            cin.ignore();
                    cin.get();
        }
        break;


        case 4:
        {
            exit=0;
        }
        break;

        default:
        {
            cout << "\033[2J\033[1;1H";
            cout<<"Invalid Input!!!"<<endl;
            cin.ignore();
                    cin.get();
        }
        break;
    }
    } while (exit);

}

int main()
{
    Interface();
```

```
}
```

Code:

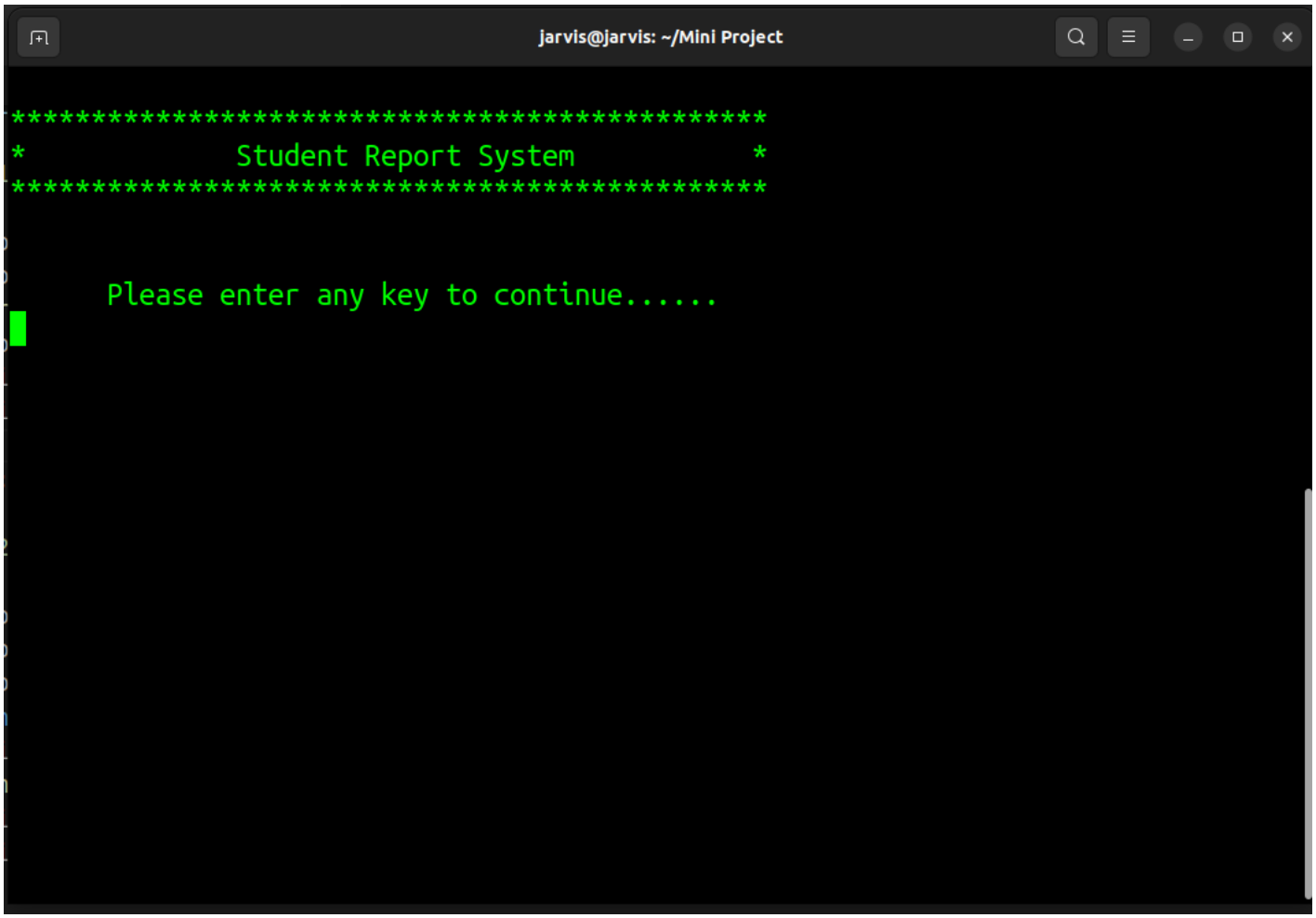
```
import java.io.*;
import java.util.*;

// class student
class student implements Serializable{
    private String name;
    private int roll_no;
    private float marks_maths; // marks in maths
    private float marks_physics; // marks in physics
    private float marks_chemistry; // marks in physics
    private float total_marks; // total marks out of 300
    private float percent; // percentage
    public void get_data()
    {
        System.out.println("Enter Name:");
        Scanner in = new Scanner(System.in);
        name = in.nextLine();
        System.out.println("Enter roll number:");
        roll_no = in.nextInt();
        System.out.println("Enter marks of physics:");
        marks_physics = in.nextFloat();
        System.out.println("Enter marks of chemistry:");
        marks_chemistry = in.nextFloat();
        System.out.println("Enter marks of maths:");
        marks_maths = in.nextFloat();
    }
    public void put_data()
    {
        System.out.println("Name:"+name);
        System.out.println("Roll Number:"+roll_no);
        System.out.println("Physics:"+marks_physics);
        System.out.println("Chemistry:"+marks_chemistry);
        System.out.println("Maths:"+marks_maths);
    }
    public void result()
    {
        System.out.println("Name:"+name);
        total_marks=marks_maths+marks_physics+marks_chemistry;
        percent=total_marks/3;
        System.out.println("Roll Number:"+roll_no);
        System.out.println("Physics:"+marks_physics);
        System.out.println("Chemistry:"+marks_chemistry);
        System.out.println("Maths:"+marks_maths);
        System.out.println("Percentage:"+percent);
    }
    public boolean check_rollno(int num)
```

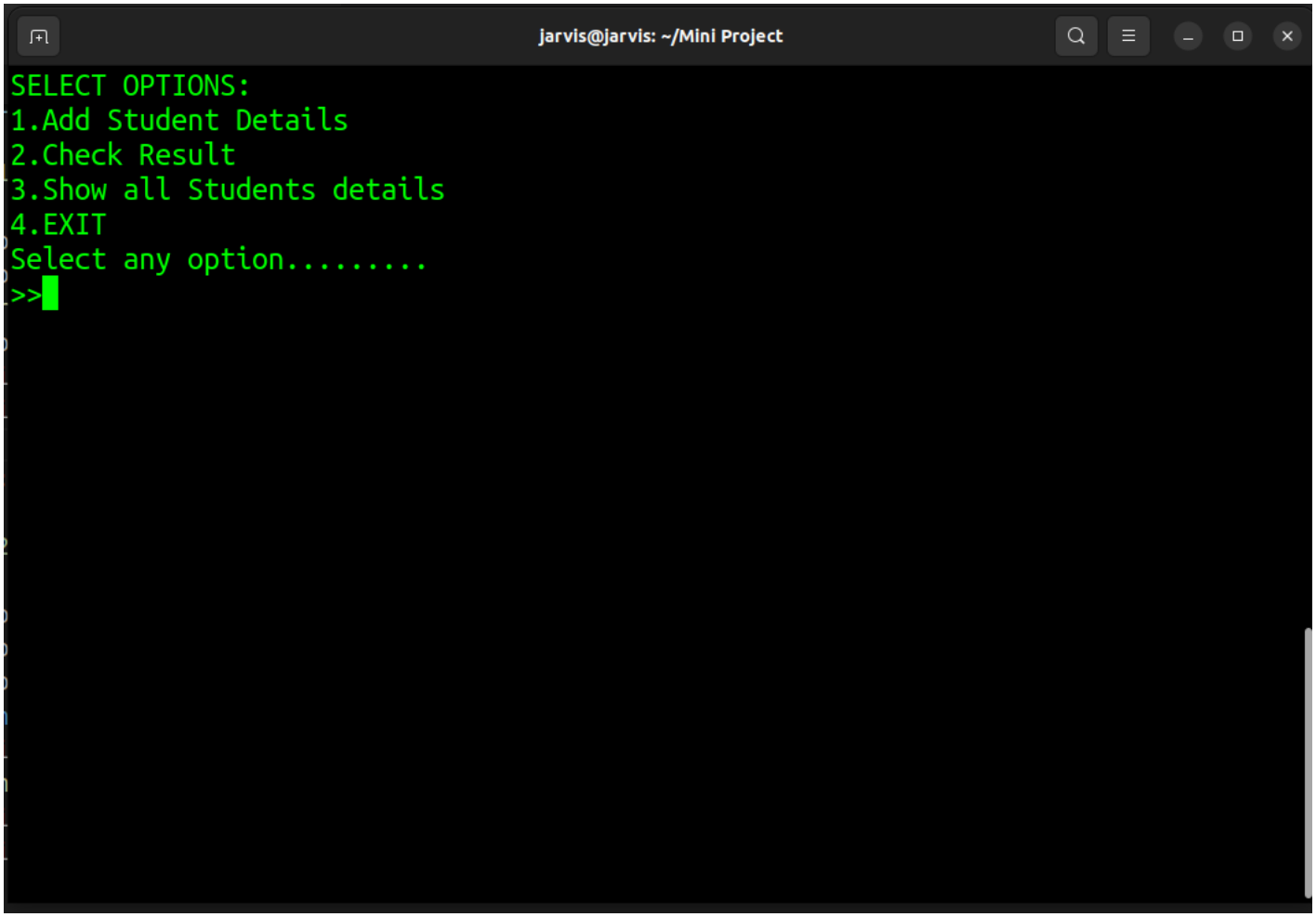
```
    {
        if(num==roll_no)
        {
            return true;
        }
        else return false;
    }
}
class Student_Report{
    public static void write_data() throws Exception
    {
        // student object s
        student s = new student();

        // object to open file and write data
        File data =new File("Student_Report.dat");

        // input data in object s
        s.get_data();

        //writing data into file
        ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(data)
        oos.writeObject(s);

        // closing file
        oos.close();

    }

    public static void show_data() throws Exception
    {
        File data = new File("Student_Report.dat");

        ObjectInputStream ois = new ObjectInputStream(new FileInputStream(data));

        boolean cond = true;

        while(cond)
        {
            student s = (student)ois.readObject();
            if(s!=null)
            {
                s.put_data();
            }
            else{
                cond=false;
            }
        }
        ois.close();
```

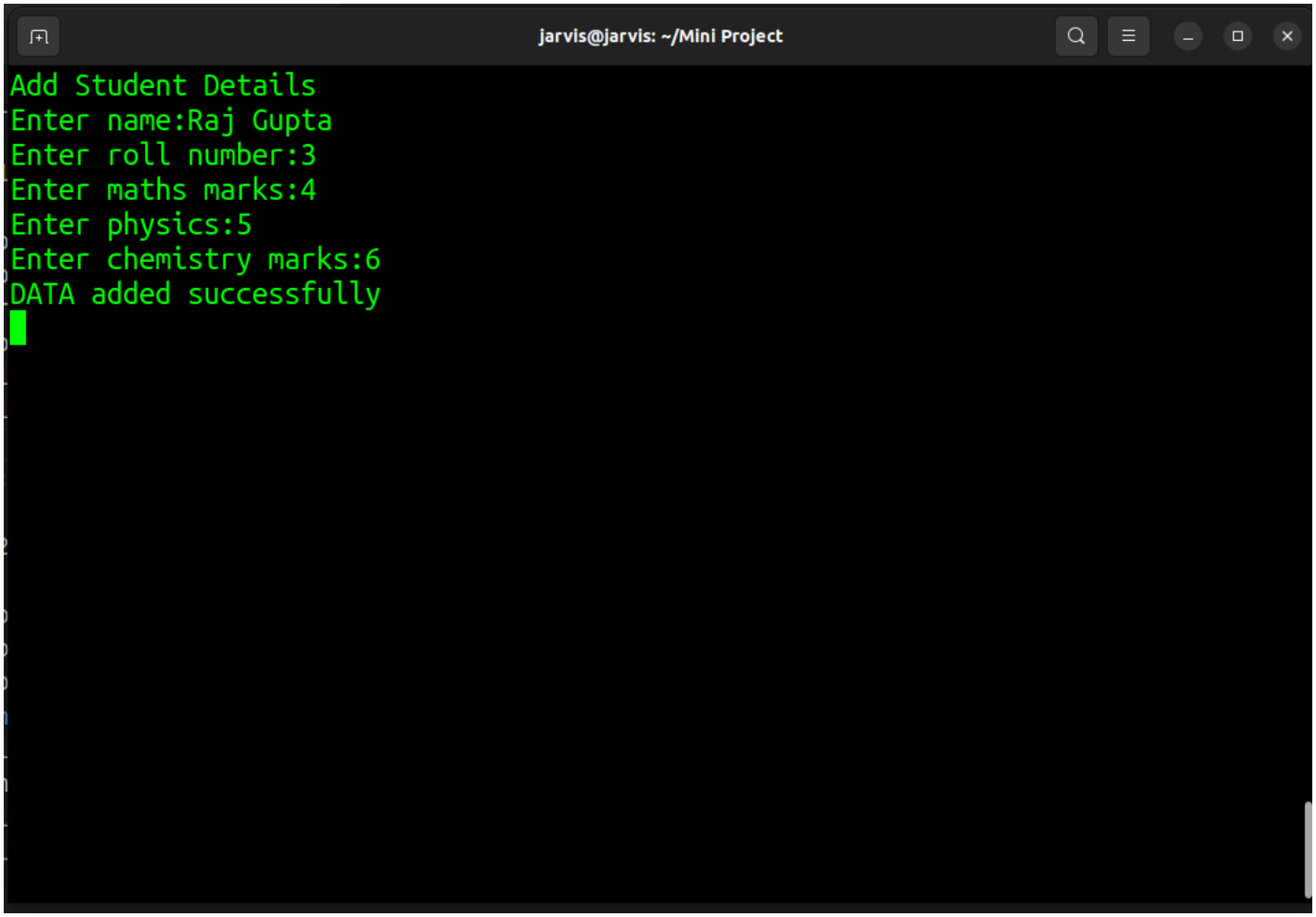
```
}

public static void show_result (int roll_no) throws Exception
{
    File data = new File("Student_Report.dat");

    ObjectInputStream ois = new ObjectInputStream(new FileInputStream(data));

    boolean cond = true;

    while(cond)
    {
        student s = (student)ois.readObject();
        if(s!=null)
        {
            if(s.check_rollno(roll_no))
            {
                s.result();
                break;
            }
        }
        else{
            cond=false;
        }
    }
    ois.close();

}
public static void screen()
{
    System.out.println("\033[2J\033[1;1H");
    System.out.println("Student Report System");
    System.out.println("Select any option:");
    System.out.println("1.Add Student Details");
    System.out.println("2.Check Result");
    System.out.println("3.Show all Students details");
    System.out.println("4.Exit");
    int option;
    Scanner input = new Scanner(System.in);
    while(true)
    {
        option = input.nextInt();
    switch(option)
    {
        case 1:
        {
            System.out.println("\033[2J\033[1;1H");
            System.out.println("Add Student details:");
```

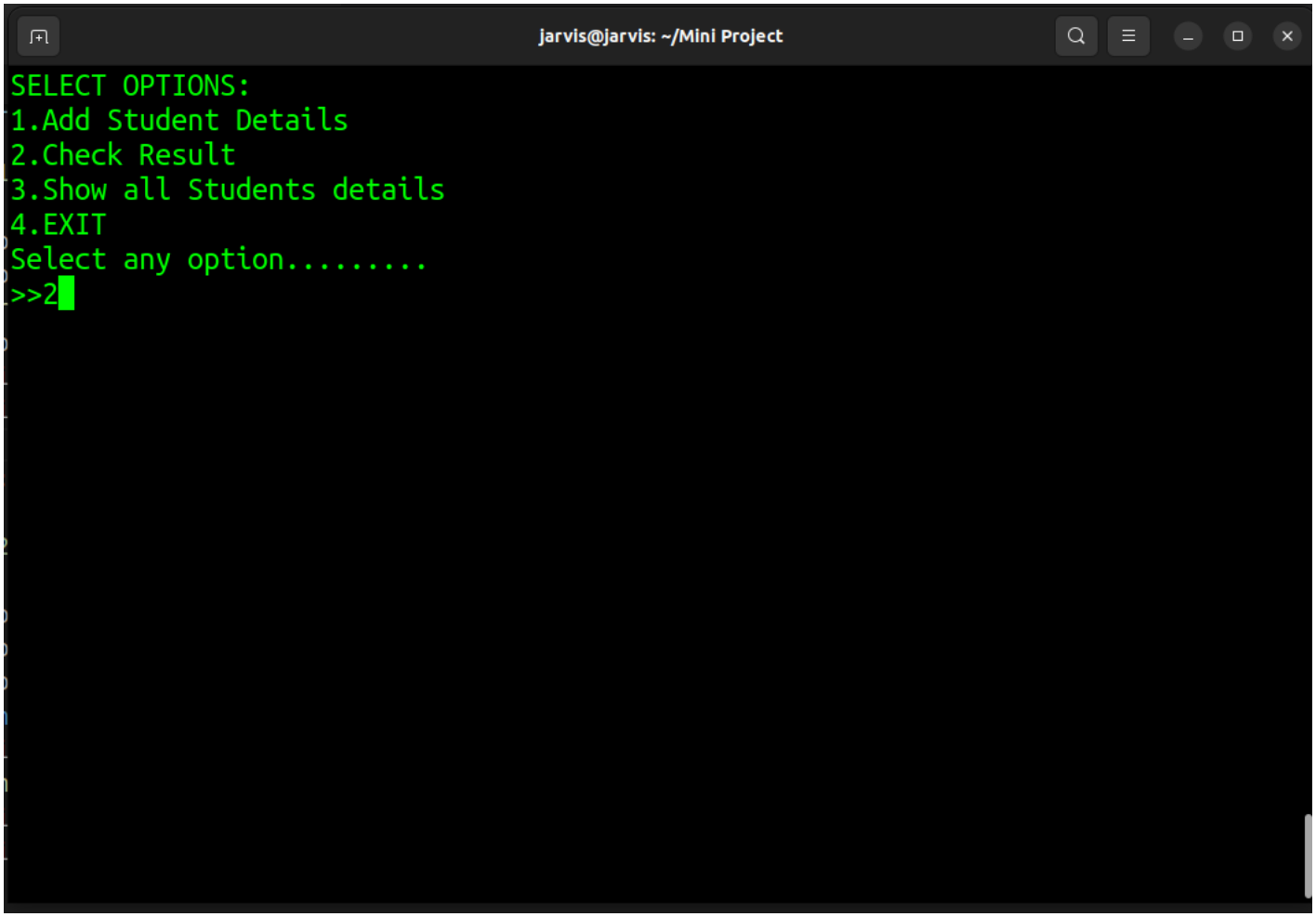
```
                write_data();
            }
            break;

            case 2:
            {
                System.out.println("\033[2J\033[1;1H");
                System.out.println("Check Result:");
                System.out.println("Enter roll number:");
                int r_no;
                Scanner in = new Scanner(System.in);
                r_no = in.nextInt();
                show_result(r_no);
            }
            break;

            case 3:
            {
                System.out.println("\033[2J\033[1;1H");
                System.out.println("Show all students results:");
                show_data();
            }
            break;

            case 4:
            {
                break;
            }
            break;

            default:
            {
                System.out.println("Invalid Input");

            }
            break;

        }
        }

    }

    public static void main(String args[])
    {
        screen();
    }
}
```

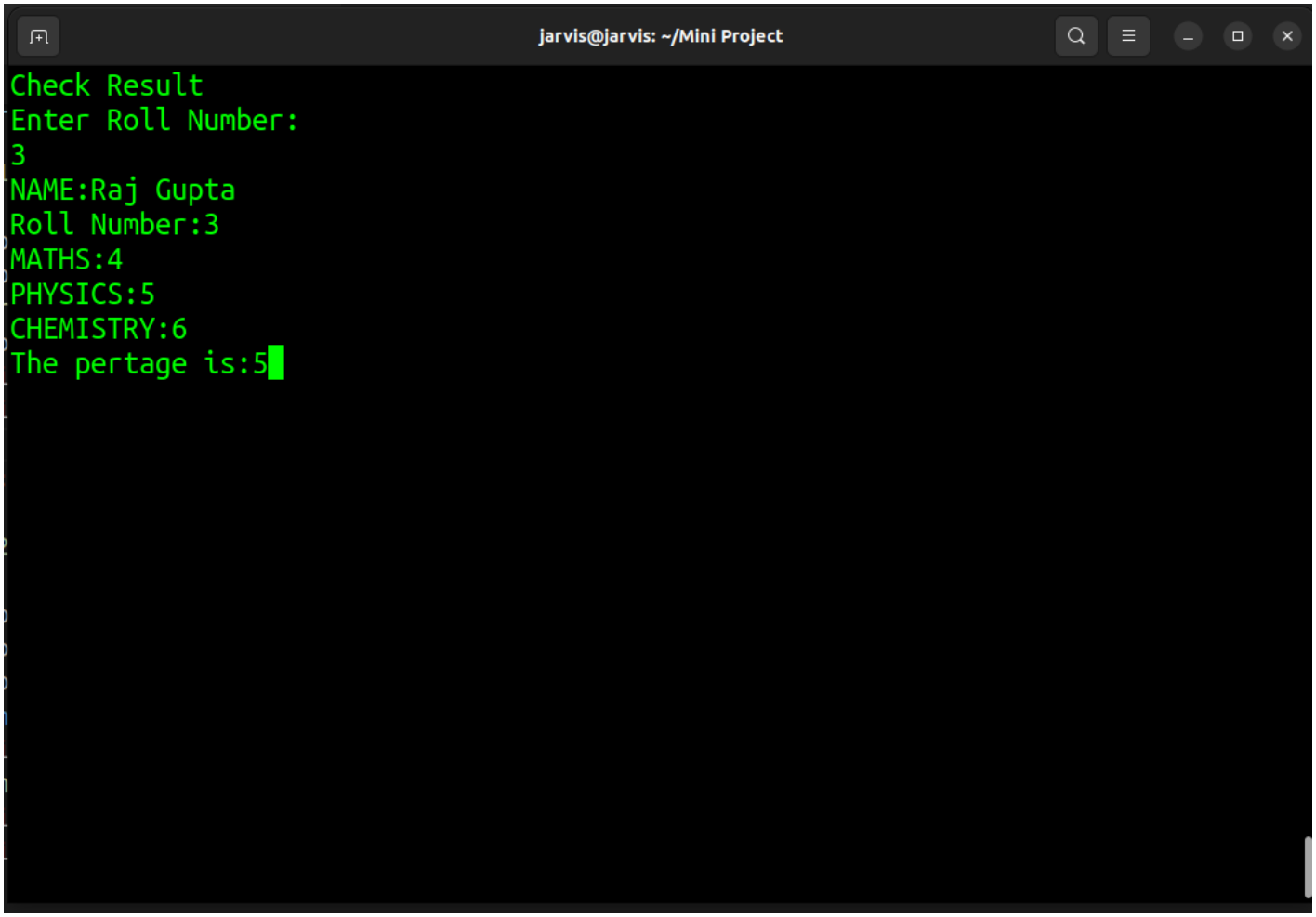
OUTPUT:

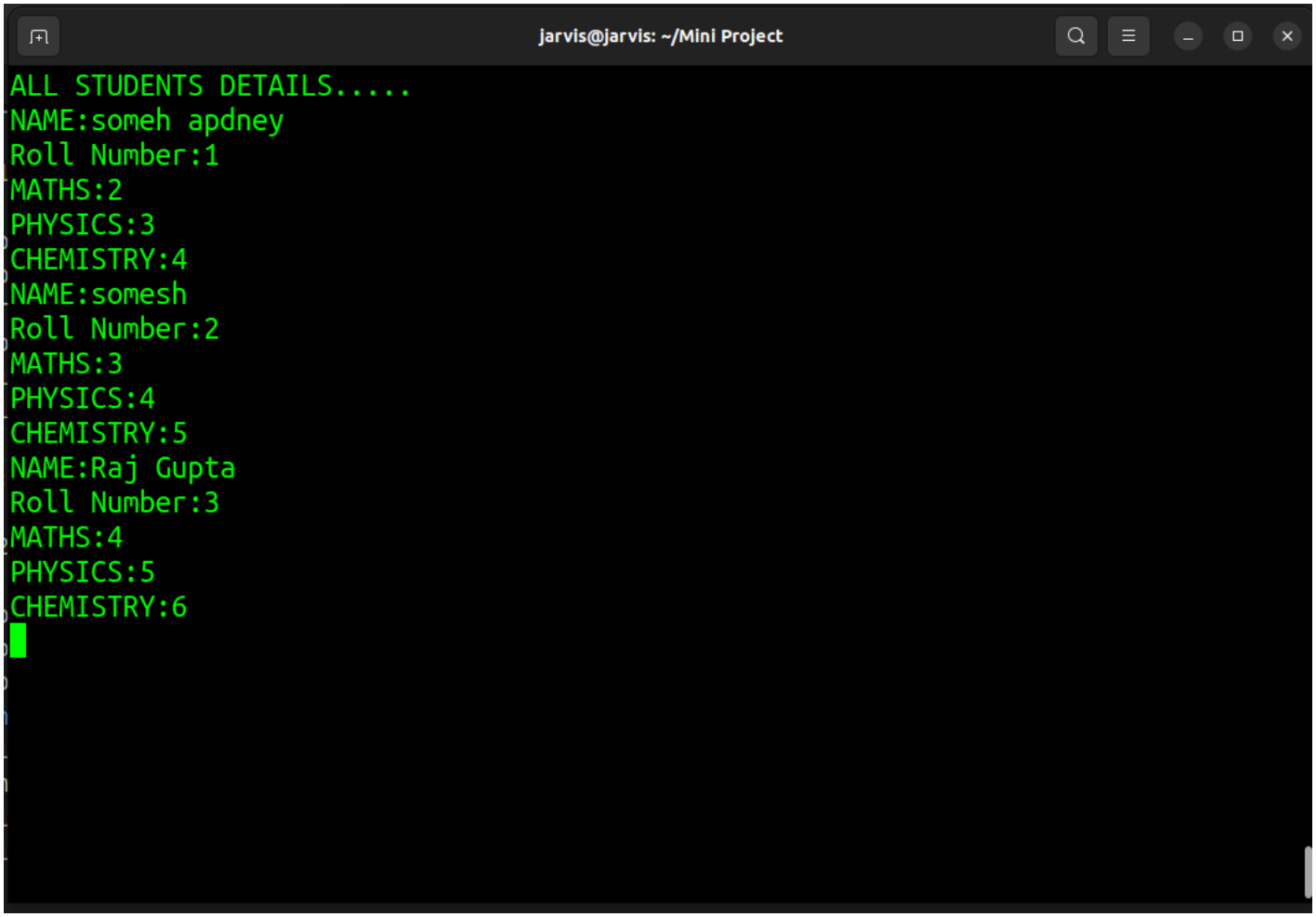

jarvis@jarvis: ~/Mini Project
****
* Student Report System *
****
Please enter any key to continue.....


jarvis@jarvis: ~/Mini Project
SELECT OPTIONS:
1.Add Student Details
2.Check Result
3.Show all Students details
4.EXIT
Select any option.....
>>


jarvis@jarvis: ~/Mini Project
Add Student Details
Enter name:Raj Gupta
Enter roll number:3
Enter maths marks:4
Enter physics:5
Enter chemistry marks:6
DATA added successfully


jarvis@jarvis: ~/Mini Project
SELECT OPTIONS:
1.Add Student Details
2.Check Result
3.Show all Students details
4.EXIT
Select any option.....
>>2


jarvis@jarvis: ~/Mini Project
Check Result
Enter Roll Number:
3
NAME:Raj Gupta
Roll Number:3
MATHS:4
PHYSICS:5
CHEMISTRY:6
The pertage is:5


jarvis@jarvis: ~/Mini Project
ALL STUDENTS DETAILS.....
NAME:someh apdney
Roll Number:1
MATHS:2
PHYSICS:3
CHEMISTRY:4
NAME:somesh
Roll Number:2
MATHS:3
PHYSICS:4
CHEMISTRY:5
NAME:Raj Gupta
Roll Number:3
MATHS:4
PHYSICS:5
CHEMISTRY:6