21BDS0340

Abhinav Dinesh Srivatsa

Structured and Object-Oriented Programming

BCSE102

Digital Assignment 1

Contents

[M1\_1 3](#_Toc103849683)

[AIM 3](#_Toc103849684)

[Algorithm / Pseudocode 3](#_Toc103849685)

[Program Code 4](#_Toc103849686)

[Output 5](#_Toc103849687)

[M1\_2 6](#_Toc103849688)

[AIM 6](#_Toc103849689)

[Algorithm / Pseudocode 6](#_Toc103849690)

[Program Code 7](#_Toc103849691)

[Output 8](#_Toc103849692)

[M2\_1 9](#_Toc103849693)

[AIM 9](#_Toc103849694)

[Algorithm / Pseudocode 9](#_Toc103849695)

[Program Code 9](#_Toc103849696)

[Output 11](#_Toc103849697)

[M2\_2 12](#_Toc103849698)

[AIM 12](#_Toc103849699)

[Algorithm / Pseudocode 12](#_Toc103849700)

[Program Code 14](#_Toc103849701)

[Output 17](#_Toc103849702)

[M3\_1 18](#_Toc103849703)

[AIM 18](#_Toc103849704)

[Algorithm / Pseudocode 18](#_Toc103849705)

[Program Code 18](#_Toc103849706)

[Output 19](#_Toc103849707)

[M4\_1 20](#_Toc103849708)

[AIM 20](#_Toc103849709)

[Algorithm / Pseudocode 20](#_Toc103849710)

[Program Code 21](#_Toc103849711)

[Output 23](#_Toc103849712)

[M4\_2 24](#_Toc103849713)

[AIM 24](#_Toc103849714)

[Algorithm / Pseudocode 24](#_Toc103849715)

[Program Code 25](#_Toc103849716)

[Output 27](#_Toc103849717)

[M5\_1 28](#_Toc103849718)

[AIM 28](#_Toc103849719)

[Algorithm / Pseudocode 28](#_Toc103849720)

[Program Code 29](#_Toc103849721)

[Output 30](#_Toc103849722)

[M5\_2 31](#_Toc103849723)

[AIM 31](#_Toc103849724)

[Algorithm / Pseudocode 31](#_Toc103849725)

[Program Code 32](#_Toc103849726)

[Output 33](#_Toc103849727)

[M6\_1 34](#_Toc103849728)

[AIM 34](#_Toc103849729)

[Algorithm / Pseudocode 34](#_Toc103849730)

[Program Code 35](#_Toc103849731)

[Output 37](#_Toc103849732)

[M6\_2 38](#_Toc103849733)

[AIM 38](#_Toc103849734)

[Algorithm / Pseudocode 38](#_Toc103849735)

[Program Code 41](#_Toc103849736)

[Output 44](#_Toc103849737)

## M1\_1

### AIM

The table below shows the normal boiling points of several substances. Write a program that prompts the user for the observed boiling point of a substance in °C and identifies the substance if the observed boiling point is within 5% of the expected boiling point. If the data input is more than 5% higher or lower than any of the boiling points in the table, the program should output the message Substance unknown.

Table

Description automatically generated

### Algorithm / Pseudocode

Declare integer array Temps as array of given temperatures

Declare integer T

Read input and store as T

Declare integer flag as 0

Loop from 0 to length of array Temps as X

Declare integer Lowert as Temps[X] \* 0.95

Declare integer Uppert as Temps[X] \* 1.05

If T > Lowert and T < Uppert, then

If X = 0, then display ‘Water’

If X = 1, then display ‘Mercury

If X = 2, then display ‘Copper

If X = 3, then display ‘Silver’

If X = 4, then display ‘Gold’

Assign Flag as 1

Break the loop

If Flag = 0, then print ‘Substance unknown’

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

int main()

{

    int temps[] = {100, 357, 1187, 2193, 2660};

    int t;

    scanf("%d", &t);

    int flag = 0;

    for (int x = 0; x < sizeof(temps) / sizeof(temps[0]); x++)

    {

        int lowert = temps[x] \* 0.95;

        int uppert = temps[x] \* 1.05;

        if (t > lowert && t < uppert)

        {

            switch (x)

            {

            case 0:

                printf("Water");

                break;

            case 1:

                printf("Mercury");

                break;

            case 2:

                printf("Copper");

                break;

            case 3:

                printf("Silver");

                break;

            case 4:

                printf("Gold");

            }

            flag = 1;

            break;

        }

    }

    if (flag == 0)

    {

        printf("Substance unknown");

    }

}

### Graphical user interface, application Description automatically generatedOutput

## M1\_2

### AIM

Develop a c Program to deal with N loans. Use math.h library for pow(a,b) and printf("%.2f",a) to print a value with two precision.

A page of a book

Description automatically generated with low confidence

### Algorithm / Pseudocode

Declare integer N

Read input and assign to N

Declare integers P, D, I, T

Loop for integer X as 0 while less than N

Read 4 inputs and assign them to P, D, I, T respectively

If T = 36, 48 or 60, then

Declare float Loan as P – D

Assign Loan as Loan \* I / 1200

Declare float Den as 1 – (1 + I/1200)-T

Assign Loan as Loan / Den

Display Loan with 2 decimal points

### Program Code

//21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

#include <math.h>

int main()

{

    int n;

    scanf("%d", &n);

    int p, d, i, t;

    for (int x = 0; x < n; x++)

    {

        scanf("%d %d %d %d", &p, &d, &i, &t);

        if (t == 36 || t == 48 || t == 60)

        {

            float loan = p - d;

            loan \*= (float)i / 1200;

            float den = 1 - pow(1 + (float)i/1200, -(float)t);

            loan /= den;

            printf("%.2f\n", floor(loan \* 100) / 100);

        }

    }

}

### Output

Graphical user interface

Description automatically generated with medium confidence

## M2\_1

### AIM

Huffman code is a particular type of optimal prefix code for characters. It is commonly used for lossless data compression. It is a variable-length code derived from frequency of occurrence. Given a string develop an algorithm and write a C program to determine frequency of occurrence of each character in the string.

### Algorithm / Pseudocode

Declare character array as Sen

Read input till new line and assign to Sen

Declare integer array Count

Declare integer X as 0

Loop while X < 26

Assign Count[X] as 0 and increment X

Declare integer Ord

Loop while Sen[X] is not ‘\0’

Calculate Ord as integer casted Sen[X]

If Ord is between ‘A’ and ‘Z’, then make it lower case

If Ord is not space, then increment Count[Ord – ‘a’]

Loop from 0 to 26 as X

If Count[X] is not 0, then display the character spaced with the Count[X]

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

int main()

{

    char sen[50];

    scanf("%[^\n]s", sen);

    int count[26];

    int x = 0;

    while (x < 26)

    {

        count[x++] = 0;

    }

    x = 0;

    int ord;

    while (sen[x] != '\0')

    {

        ord = (int)sen[x++];

        if (ord <= 'Z' && ord >= 'A')

        {

            ord += 32;

        }

        if (ord != ' ')

        {

            count[ord - 'a']++;

        }

    }

    for (x = 0; x < 26; x++)

    {

        if (count[x] != 0)

        {

            printf("%c %d\n", x + 'a', count[x]);

        }

    }

}

### Output

Graphical user interface, application

Description automatically generated

## M2\_2

### AIM

A picture containing text, antenna

Description automatically generatedTic-tac-toe is a paper-and-pencil game for two players, X and O, who take turns marking the spaces n a 3×3 grid. Player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game.

Given the board configuration of the tic tac toe game, determine if the board is in either of the following states: empty, player1 wins, player2 wins, draw or intermediate. The board is said to be in initial state if all the cells contain ‘-1’, player1 uses ‘1’ as his coin and player2 uses ‘2’ as his coin. The game is draw when the board is full and no one has won the game. The game is in intermediate state when no one has won and board is not full

Use the following function signatures

void read\_Board(int ttt[][3]);

int count\_EmptyCell(int ttt[][3]);

int check\_Rowwise(int ttt[][3],int);

int check\_Colwise(int ttt[][3],int);

int check\_Diagonalwise(int ttt[][3],int);

### Algorithm / Pseudocode

Void read\_Board(int TTT[3][3])

Loop from 0 to 3 as X

Loop from 0 to 3 as Y

Read input and assign to TTT[X][Y]

Int count\_EmptyCell(int TTT[3][3])

Declare integer Count as 0

Loop from 0 to 3 as X

Loop from 0 to 3 as Y

If TTT[X][Y] = -1, then increment Count

Return Count

Int check\_Rowwise(int TTT[3][3])

Declare integer Val

Loop from 0 to 3 as X

Assign Val as TTT[X][0]

If TTT[X][1] and TTT[X][2] are equal to Val, then return Val

Return 0

Int check\_Colwise(int TTT[3][3])

Declare integer Val

Loop from 0 to 3 as X

Assign Val as TTT[0][X]

If TTT[1][X] and TTT[2][X] are equal to Val, then return Val

Return 0

Int check\_Diagonalwise(int TTT[3][3])

Declare integer Val

Assign Val as TTT[0][0]

If TTT[1][1] and TTT[2][2] equal Val, then return Val

Assign Val as TTT[0][2]

If TTT[1][1] and TTT[2][0] equal Val, then return Val

Return 0

Declare integer array TTT with 3 rows and 3 columns

Call read\_Board(TTT)

Declare integer Emptycells and assign it by calling count\_EmptyCell(TTT)

If Emptycells = 9, then display ‘Empty’

Else if Emptycells = 0, then

Declare integer Rowwin and assign it by calling check\_Rowwise(TTT)

Declare integer Colwin and assign it by calling check\_Colwise(TTT)

Declare integer Diagwin and assign it by calling check\_Diagonalwise(TTT)

If Rowwin, Colwin and Diagwin are 0, then display ‘Draw’

Else if Rowwin is not 0, then display ‘Player{Rowwin} wins’

Else if Colwin is not 0, then display ‘Player{Colwin} wins’

Else if Diagwin is not 0, then display ‘Player{Diagwin} wins’

Else display ‘Intermediate’

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

void read\_Board(int ttt[3][3])

{

    for (int x = 0; x < 3; x++)

    {

        for (int y = 0; y < 3; y++)

        {

            scanf("%d", &ttt[x][y]);

        }

    }

}

int count\_EmptyCell(int ttt[3][3])

{

    int count = 0;

    for (int x = 0; x < 3; x++)

    {

        for (int y = 0; y < 3; y++)

        {

            if (ttt[x][y] == -1)

            {

                count++;

            }

        }

    }

    return count;

}

int check\_Rowwise(int ttt[3][3])

{

    int val;

    for (int x = 0; x < 3; x++)

    {

        val = ttt[x][0];

        if (ttt[x][1] == val && ttt[x][2] == val)

        {

            return val;

        }

    }

    return 0;

}

int check\_Colwise(int ttt[3][3])

{

    int val;

    for (int x = 0; x < 3; x++)

    {

        val = ttt[0][x];

        if (ttt[1][x] == val && ttt[2][x] == val)

        {

            return val;

        }

    }

    return 0;

}

int check\_Diagonalwise(int ttt[3][3])

{

    int val;

    // primary diagonal

    val = ttt[0][0];

    if (ttt[1][1] == val && ttt[2][2] == val)

    {

        return val;

    }

    // secondary diagonal

    val = ttt[0][2];

    if (ttt[1][1] == val && ttt[2][0] == val)

    {

        return val;

    }

    return 0;

}

int main()

{

    int ttt[3][3];

    read\_Board(ttt);

    int emptycells = count\_EmptyCell(ttt);

    if (emptycells == 9)

    {

        printf("Empty");

    }

    else if (emptycells == 0)

    {

        int rowwin = check\_Rowwise(ttt);

        int colwin = check\_Colwise(ttt);

        int diagwin = check\_Diagonalwise(ttt);

        if (rowwin == 0 && colwin == 0 && diagwin == 0)

        {

            printf("Draw");

        }

        else if (rowwin != 0)

        {

            printf("Player%d wins", rowwin);

        }

        else if (colwin != 0)

        {

            printf("Player%d wins", colwin);

        }

        else if (diagwin != 0)

        {

            printf("Player%d wins", diagwin);

        }

    }

    else

    {

        printf("Intermediate");

    }

}

### Output

Graphical user interface

Description automatically generated

## M3\_1

### AIM

Create a program in C to dynamically allocate integer array. Display the elements of the array using dereferencing operator in the reverse order.

### Algorithm / Pseudocode

Declare integer N

Read input and assign to N

Declare integer pointer Arr and assign array of N spaces

Loop from 0 to N as X

Read input and assign to Arr[X]

Calculate Arr as Arr + N - 1

Loop from N - 1 to -1 as X

Display value of Arr

Decrement Arr

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

#include <stdlib.h>

int main()

{

    int n;

    scanf("%d", &n);

    int \*arr = malloc(n \* sizeof(int));

    for (int x = 0; x < n; x++)

        scanf("%d", (arr + x));

    for (int x = n - 1; x > -1; x--)

        printf("%d\n", \*(arr + x));

    free(arr);

}

### Output

Graphical user interface

Description automatically generated

## M4\_1

### AIM

Create an employee structure with elements, empid, name, age, dept, designation, salary. Define array of employees and pass to a function to read the values and another function to sort the employees based on age attribute, display only empid.

### Algorithm / Pseudocode

Declare structure employee with 6 fields:

    Integer Empid

    Character array Name with 10 spaces

    Integer Age

    Character array Dept with 6 spaces

    Character array Desig with 15 spaces

    Integer Salary

void sort\_21BDS0340(struct employee \*E)

    Read inputs and store as E's Empid, Name, Age, Dept, Desig, Salary

void sort\_21BDS0340(struct employee E[3])

    Declare employee Temp

    Loop from 0 to 2 as X

    Loop from 0 to 2 - X as Y

        If E[Y]'s Age > E[Y+1]'s Age, then

                Assign Temp as E[Y]

                Assign E[Y] as E[Y+1]

                Assign E[Y+1] as Temp

Declare employee E with 3 spaces

Loop from 0 to 3 as X

    Call getValues\_21BDS0340 and pass E[X]

Call sort\_21BDS0340 and pass E

Loop from 2 to -1 as X

    Display E[X]'s Empid

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

struct employee

{

    int empid;

    char name[10];

    int age;

    char dept[6];

    char desig[15];

    int salary;

};

void getValues\_21BDS0340(struct employee \*e)

{

    scanf("%d%s%d%s%s%d", &e->empid, e->name, &e->age, e->dept, e->desig, &e->salary);

}

void sort\_21BDS0340(struct employee e[3])

{

    struct employee temp;

    for (int x = 0; x < 2; x++)

        for (int y = 0; y < 2 - x; y++)

            if (e[y].age > e[y + 1].age)

            {

                temp = e[y];

                e[y] = e[y + 1];

                e[y + 1] = temp;

            }

}

int main()

{

    struct employee e[3];

    for (int x = 0; x < 3; x++)

    {

        getValues\_21BDS0340(&e[x]);

    }

    sort\_21BDS0340(e);

    for (int x = 2; x >= 0; x--)

    {

        printf("%d\n", e[x].empid);

    }

}

### Output

Graphical user interface

Description automatically generated

## M4\_2

### AIM

The weather station of each city has the detail of rainfall in a year. Given the date and cm of rainfall recorded on that day, write a C program to determine the rainfall recorded in each month of the year and average monthly rainfall in the year

Note: please use structure for rainfall and date with necessary attributes. Please store date character array appropriately in date structure variable while returning from function.

### Algorithm / Pseudocode

Declare structure date with 3 fields:

Integer D

Integer M

Integer Y

Declare structure rain with 2 fields:

Date Date

Integer Cm

struct date dateToStruct(char Array[11])

Declare date D

Assign D.D as the date part of Array

Assign D.M as the month part of Array

Assign D.Y as the year part of Array

Return D

Declare integer N

Read input and assign to N

Declare rain array R with N spaces

Declare character array Datestr with 11 spaces

Loop from 0 to N as X

Read inputs and assign to Datestr and R[X]'s Cm

Assign R[X]'s Date by calling dateToStruct and passing Datestr

Declare integer array Month\_rain with 12 empty spaces

Declare integer Total\_rain as 0

Loop from 0 to N as X

Calculate Month\_rain[R[X]'s Date's D] as itself + R[X]'s Cm

Calculate Total\_rain as itself + R[X]'s Cm

Loop from 0 to N as X

If Month\_rain[X] is not 0, then display X + 1 and Month\_rain[X]

Display Total\_rain / N

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

struct date

{

    int d;

    int m;

    int y;

};

struct rain

{

    struct date date;

    int cm;

};

struct date dateToStruct(char array[11])

{

    struct date d;

    d.d = atoi(array);

    d.m = atoi(&array[3]);

    d.y = atoi(&array[6]);

    return d;

}

int main()

{

    int n;

    scanf("%d", &n);

    struct rain \*r = malloc(n \* sizeof(struct rain));

    char datestr[11];

    for (int x = 0; x < n; x++)

    {

        scanf("%s%d", datestr, &(r + x)->cm);

        (r + x)->date = dateToStruct(datestr);

    }

    int \*month\_rain = calloc(12, sizeof(int));

    int total\_rain = 0;

    for (int x = 0; x < n; x++)

    {

        month\_rain[(r + x)->date.m - 1] += (r + x)->cm;

        total\_rain += (r + x)->cm;

    }

    for (int x = 0; x < 12; x++)

        if (month\_rain[x] != 0)

            printf("%d\n%d\n", (x + 1), month\_rain[x]);

    printf("%d", (int)round((float)total\_rain / n));

    free(month\_rain);

    free(r);

}

### Output

Graphical user interface, application

Description automatically generated

## M5\_1

### AIM

In an online examination system, each test will be scheduled for ‘x’ minutes. The student is free to take up the test on his convenience but once he starts the test, he must complete. Given the start time and the value of ‘x’ for an examination, develop an algorithm and write a ‘C++’ code for the examination system to calculate the finish time of the test

### Algorithm / Pseudocode

Test:

Declare integer Testtime

Declare integer Hour

Declare integer Minute

Public void readValues():

Declare integer TT, H, M

Assign Testtime as TT

Assign Hour as H

Assign Minute as M

Public void showEndTime():

Calculate Minute as Minute + Testtime

If Minute > 59, then calculate Hour as Hour + Minute / 60 and calculate minute as minute % 60

If Hour > 12, then calculate Hour as Hour - 12

If Hour < 10, then display "0" + Hour + ":"

Else display Hour + ":"

If Minute < 10, then display "0" + Minute

Else display Minute

Declare Test T

Call T.readValues()

Call T.showEndTIme()

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <iostream>

using namespace std;

class Test

{

    int testtime;

    int hour;

    int minute;

public:

    void readValues()

    {

        int tt, h, m;

        cin >> h >> m >> tt;

        testtime = tt;

        hour = h;

        minute = m;

    }

    void showEndTime()

    {

        minute += testtime;

        if (minute > 59)

        {

            hour += minute / 60;

            minute %= 60;

        }

        if (hour > 12)

            hour -= 12;

        if (hour < 10)

            cout << "0" << hour << ":";

        else

            cout << hour << ":";

        if (minute < 10)

            cout << "0" << minute;

        else

            cout << minute;

    }

};

int main()

{

    Test t;

    t.readValues();

    t.showEndTime();

}

### Output

Graphical user interface, application

Description automatically generated

## M5\_2

### AIM

Develop a friend function to calculate total fare for a ticket. There are ‘n’ passengers in the ticket and they are from a family of a railway employee. Discount is given for their travel based on the cader of the employee.

If Cader A - 10%, B - 15%, C - 20%

### Algorithm / Pseudocode

Fare:

Declare integer Passengers

Declare character array From with 20 spaces

Declare character array To with 20 spaces

Declare integer Ticket\_cost

Declare character Cader

Fare(int P, char F[20], char T[20], int Tc, char C):

Assign Passengers as P

Assign From as F

Assign To as T

Assign Ticket\_cost as Tc

Assign Cader as C

Declare Price as friend class

Price:

Int calculate(Fare F):

Switch F's Cader

If 'A', then return F's Ticket\_cost \* F's Passengers \* 0.9

If 'B', then return F's Ticket\_cost \* F's Passengers \* 0.85

If 'C', then return F's Ticket\_cost \* F's Passengers \* 0.8

Return 0

Int main():

Declare integers P and Tc

Declare character C and arrays Fr and T with 20 spaces

Read input and assign to P, Fr, T, Tc and C respectively

Declare Fare F() passing P, Fr, T, Tc and C as arguments

Declare Price Pr

Call Pr's calculate() and pass F and display

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <iostream>

#include <string.h>

using namespace std;

class Fare

{

    int passengers;

    char from[20];

    char to[20];

    int ticket\_cost;

    char cader;

public:

    Fare(int p, char f[20], char t[20], int tc, char c)

    {

        this->passengers = p;

        strcpy(this->from, f);

        strcpy(this->to, t);

        this->ticket\_cost = tc;

        this->cader = c;

    }

    friend class Price;

};

class Price

{

public:

    int calculate(Fare f)

    {

        switch (f.cader)

        {

        case 'A':

            return f.ticket\_cost \* f.passengers \* 0.9;

        case 'B':

            return f.ticket\_cost \* f.passengers \* 0.85;

        case 'C':

            return f.ticket\_cost \* f.passengers \* 0.8;

        }

        return 0;

    }

};

int main()

{

    int p, tc;

    char fr[20], t[20], c;

    cin >> p >> fr >> t >> tc >> c;

    Fare f(p, fr, t, tc, c);

    Price pr;

    cout << pr.calculate(f);

}

### Output

Shape

Description automatically generated

## M6\_1

### AIM

Create a class named person with attributes name, age, phno with getPerson and printPerson member functions. Extend the class named employee with additional attributes empId, designation along with getEmp and displayEmp member functions. Define an array of employee instances and invoke the respective employee member functions and display only empId of all employees in ascending order.

### Algorithm / Pseudocode

Person:

Declare character array Name with 20 spaces

Declare integer Age

Declare character array Phno with 11 spaces

Void getPerson():

Read inputs and assign to Name, Age and Phno

Void printPerson():

Display Name, Age and Phno

Employee, extends Person:

Declare character array Desig with 20 spaces

Declare integer EmpId

Void getEmp():

Call Person's getPerson()

Read input and assign to EmpId and Desig

Void dispEmp():

Call Person's printPerson()

Display EmpId and Desig

Int main():

Declare integer N

Read input and assign as N

Declare Employee array E with N spaces

Loop from 0 to N as X

Call E[X]'s getEmp()

Loop from 0 to N as X

Loop from X to N as Y

If E[X]'s EmpId > E[Y]'s EmpId, then swap them

Loop from 0 to N as X

Display E[X]'s EmpId

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <iostream>

using namespace std;

class Person

{

    char name[20];

    int age;

    char phno[11];

public:

    void getPerson()

    {

        cin >> this->name >> this->age >> this->phno;

    }

    void printPerson()

    {

        cout << this->name << "\n"

             << this->age << "\n"

             << this->phno << "\n";

    }

};

class Employee : private Person

{

    char desig[20];

public:

    int empId;

    void getEmp()

    {

        Person::getPerson();

        cin >> this->empId >> this->desig;

    }

    void dispEmp()

    {

        Person::printPerson();

        cout << this->empId << "\n"

             << this->desig << "\n";

    }

};

int main()

{

    int n;

    cin >> n;

    Employee e[n];

    for (int x = 0; x < n; x++)

        e[x].getEmp();

    for (int x = 0; x < n; x++)

        for (int y = x + 1; y < n; y++)

            if (e[x].empId > e[y].empId)

            {

                Employee temp = e[y];

                e[y] = e[x];

                e[x] = temp;

            }

    for (int x = 0; x < n; x++)

        cout << e[x].empId << "\n";

}

### Output

Chart, treemap chart

Description automatically generated

## M6\_2

### AIM

Define a class named person[name, age, getPerson, displayPerson], extend the class and define a Student class[regno, cgpa, getStudent, displayStudent].

Create a class named GateScore[GateRegistrationNumber, Score, ExamCode, getGate, displayGate].

Develop a class PG\_Student[deptName, getPGS, displayPGS] inherited from two parents, Student class and GateScore class.

In main method, create n instance of PG students and invoke all member functions. Find the student and display VIT Regno and GATE regno, who has scored highest Gate Score.

### Algorithm / Pseudocode

Person:

Declare character array Name with 20 spaces

Declare integer Age

Void getPerson():

Read inputs and assign to Name and Age

Void displayPerson():

Display Name and Age

Student, extends Person:

Declare character array Regno with 10 spaces

Declare float Cgpa

Void getStudent():

Call getPerson()

Read inputs and assign to Regno and Cgpa

Void displayStudent():

Call displayPerson()

Display Regno and Cgpa

Char\* getRegno():

Return Regno

GateScore:

Declare character array GateRegistrationNumber with 20 spaces

Declare integer Score

Declare character array ExamCode with 20 spaces

Void getGate():

Read inputs and assign to GateRegistrationNumber, Score and ExamCode

Void displayGate():

Display GateRegistrationNumber, Score and ExamCode

Char\* getGateRegno():

Return GateRegistrationNumber

Int getGateScore():

Return Score

PG\_Student, extends Student and GateScore:

Declare character array DeptName with 10 spaces

Void getPGS():

Call getStudent()

Call getGate()

Read input and assign to DeptName

Void displayPGS():

Call displayStudent()

Call displayGate()

Display DeptName

Int main():

Declare integer N

Read input and assign to N

Declare PG\_Student array Pgs with N spaces

Loop from 0 to N as X

Call Pgs[X]'s getPGS()

Declare integer Max as Pgs[0]'s getGateScore()

Declare character arrays Regno and GateRegno with 10 and 20 spaces

Assign Regno as Pgs[0]'s getRegno()

Assign GateRegno as Pgs[0]'s getGateRegno()

Loop from 0 to N as X

If Pgs[X]'s getGateScore() > Max, then

Assign Max as Pgs[X]'s getGateScore()

Assign Regno as Pgs[X]'s getRegno()

Assign GateRegno as Pgs[X]'s getGateRegno()

Display Regno and GateRegno

### Program Code

// 21BDS0340 Abhinav Dinesh Srivatsa

#include <iostream>

#include <string.h>

using namespace std;

class Person

{

    char name[20];

    int age;

public:

    void getPerson()

    {

        cin >> name >> age;

    }

    void displayPerson()

    {

        cout << name << "\n"

             << age << "\n";

    }

};

class Student : public Person

{

    char regno[10];

    float cgpa;

public:

    void getStudent()

    {

        getPerson();

        cin >> regno >> cgpa;

    }

    void displayStudent()

    {

        displayPerson();

        cout << regno << "\n"

             << cgpa << "\n";

    }

    char \*getRegno()

    {

        return this->regno;

    }

};

class GateScore

{

    char GateRegistrationNumber[20];

    int Score;

    char ExamCode[20];

public:

    void getGate()

    {

        cin >> GateRegistrationNumber >> Score >> ExamCode;

    }

    void displayGate()

    {

        cout << GateRegistrationNumber << "\n"

             << Score << "\n"

             << ExamCode << "\n";

    }

    char \*getGateRegno()

    {

        return this->GateRegistrationNumber;

    }

    int getGateScore()

    {

        return this->Score;

    }

};

class PG\_Student : public Student, public GateScore

{

    char deptName[10];

public:

    void getPGS()

    {

        getStudent();

        getGate();

        cin >> deptName;

    }

    void displayPgs()

    {

        displayStudent();

        displayGate();

        cout << deptName << "\n";

    }

};

int main()

{

    int n;

    cin >> n;

    PG\_Student pgs[n];

    for (int x = 0; x < n; x++)

        pgs[x].getPGS();

    int max = pgs[0].getGateScore();

    char regno[10], gateRegno[20];

    strcpy(regno, pgs[0].getRegno());

    strcpy(gateRegno, pgs[0].getGateRegno());

    for (int x = 0; x < n; x++)

        if (pgs[x].getGateScore() > max)

        {

            max = pgs[x].getGateScore();

            strcpy(regno, pgs[x].getRegno());

            strcpy(gateRegno, pgs[x].getGateRegno());

        }

    cout << regno << "\n"

         << gateRegno << "\n";

}

### Output

Chart, bar chart

Description automatically generated