



VIT[®]
BHOPAL
www.vitbhopal.ac.in

Programme: B. Tech in CSE (Core)

Course Code and Title: Object Oriented Programming
with C++ CSE2001

LAB-Practical

Submitted by: Shashwat Shandilya

Registration No: 21BCE11173

Submitted to: Divya Mam

Date: 25/4/22

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENTS:

EXPERIMENT 1:

Q 1.) Design a class to represent a bank account. Include the following members.

Data Members:

- Name of the depositor
- Account number
- Type of account
- Balance amount in the account
- Methods
 - To assign initial values
 - To deposit an amount
 - To withdraw an amount after checking balance
 - To display the name and balance

Incorporate a constructor to provide initial values.

CODE:-

```
#include <iostream>
#include <stdio.h>
#include <string.h>
using namespace std;

class bank
{
    int acno;
    char nm[100], acctype[100];
    float bal;
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
public:
    bank(int acc_no, char *name, char *acc_type, float balance) // Parameterized
    Constructor
    {
        acno = acc_no;
        strcpy(nm, name);
        strcpy(acctype, acc_type);
        bal = balance;
    }
    void deposit();
    void withdraw();
    void display();
};

void bank::deposit() // depositing an amount
{
    int damt1;
    cout << "\n Enter Deposit Amount = ";
    cin >> damt1;
    bal += damt1;
}

void bank::withdraw() // withdrawing an amount
{
    int wamt1;
    cout << "\n Enter Withdraw Amount = ";
    cin >> wamt1;
    if (wamt1 > bal)
        cout << "\n Cannot Withdraw Amount";
    bal -= wamt1;
}

void bank::display() // displaying the details
{
    cout << "\n -----";
    cout << "\n Accout No. : " << acno;
    cout << "\n Name : " << nm;
    cout << "\n Account Type : " << acctype;
    cout << "\n Balance : " << bal;
    cout << "\n -----";
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
int main()
{
    int acc_no;
    char name[100], acc_type[100];
    float balance;
    cout << "\n Enter Details: \n";
    cout << "-----";
    cout << "\n Account No. : ";
    cin >> acc_no;
    cout << "\n Name : ";
    cin >> name;
    cout << "\n Account Type : ";
    cin >> acc_type;
    cout << "\n Balance : ";
    cin >> balance;
    cout << "-----";
    bank b1(acc_no, name, acc_type, balance); // object is created
    b1.deposit();                             //
    b1.withdraw();                             // calling member functions
    b1.display();                             //
    return 0;
}
```

OUTPUT:

```
Enter Details:
-----
Account No. : 45973

Name : Shashwat

Account Type : Savings

Balance : 500000
-----
Enter Deposit Amount = 50000

Enter Withdraw Amount = 10000

-----
Account No. : 45973
Name : Shashwat
Account Type : Savings
Balance : 540000
-----
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 2:

Q 2.) Guess-the-number-game: Write a program that plays the game of “guess the number” as follows:
Your program chooses the number to be guessed by selecting an integer at random in the range 1 to 1000. The program then displays the following:

- a) I have a number between 1 and 1000.
 - b) Can you guess my number?
 - c) Please type your first guess.
 - d) The player then types a first guess. The program responds with one of the following:
 - Excellent ! you guessed the number! Would like to play again (y or n)?
 - Too low. Try again.
 - Too high. Try again.
 - e) If the player's guess is incorrect, your program should loop until the player finally gets the number right. Your program should keep telling the player Too high Too low to help the player “zero in” on the correct answer.
-

CODE:-

```
#include <iostream>
using std::cin;
using std::cout;
using std::endl;
#include <cstdlib>
#include <ctime>

int main()
{
    srand(time(0));
    int guess;
    int number;
    char selection = 'y';
    while (selection == 'y')
    {
        number = rand() % 1000 + 1;
        cout << "\n(Hint: Number is " << number << ")." << endl; // check the
        program for cheating
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        cout<< "\n\n I have a number between 1 and 1000.\n Can you guess my
number?\n Please type your first guess: ";
        cin >> guess;

        while (number != guess)
        {
            if (number > guess)
            {
                cout << "\n Too low. Try again." << endl;
                cin >> guess;
            }
            if (number < guess)
            {
                cout << "\n Too high. Try again." << endl;
                cin >> guess;
            }
        }
        cout << "\n\n Excellent! You guessed the number!\n Would you like to play
again (y or n)? ";

        cin >> selection;
    }

    cout << "\n\n Thank you!";
    return 0;
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:

```
(Hint: Number is 425).
```

```
I have a number between 1 and 1000.  
Can you guess my number?  
Please type your first guess: 100
```

```
Too low. Try again.  
200
```

```
Too low. Try again.  
400
```

```
Too low. Try again.  
500
```

```
Too high. Try again.  
450
```

```
Too high. Try again.  
440
```

```
Too high. Try again.  
420
```

```
Too low. Try again.  
424
```

```
Too low. Try again.  
425
```

```
Excellent! You guessed the number!  
would you like to play again (y or n)? n
```

```
Thank you!
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 3:

Q 3.) Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The savings account provides simple interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class account that stores customer name, account number and type of account. From this derive the classes cur_acct and sav_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:

- Accept deposit from a customer and update the balance.
 - Display the balance.
 - Compute and deposit interest.
 - Permit withdrawal and update the balance.
 - Check for the minimum balance, impose penalty, necessary and update the balance.
- Do not use any constructors. Use member functions to initialize the class members.

CODE:-

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

class account
{
private:
    string name;
    int accno;
    string atype;

public:
    void getAccountDetails()
    {
        cout << "\nEnter Customer Name : ";
```


Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        cin >> name;
        cout << "Enter Account Number  : ";
        cin >> accno;
        cout << "Enter Account Type    : ";
        cin >> atype;
    }
    void displayDetails()
    {
        cout << "\n\nCustomer Name : " << name;
        cout << "\nAccount Number : " << accno;
        cout << "\nAccount Type    : " << atype;
    }
};
class current_account : public account
{
private:
    float balance = 5000;
public:
    void c_display()
    {
        cout << "\nBalance :" << balance;
    }
    void c_deposit()
    {
        float deposit;
        cout << "\nEnter amount to Deposit : ";
        cin >> deposit;
        balance = balance + deposit;
    }
    void c_withdraw()
    {
        float withdraw;
        cout << "\n\nBalance : " << balance;
        cout << "\nEnter amount to be withdraw :";
        cin >> withdraw;
        if (balance > 1000)
        {
            balance = balance - withdraw;
            cout << "\nBalance Amount After Withdraw: " << balance;
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
    }
    else
    {
        cout << "\n Insufficient Balance";
    }
}
};

class saving_account : public account
{
private:
    float sav_balance = 5000;

public:
    void s_display()
    {
        cout << "\nBalance : " << sav_balance;
    }
    void s_deposit()
    {
        float deposit, interest;
        cout << "\nEnter amount to Deposit : ";
        cin >> deposit;
        sav_balance = sav_balance + deposit;
        interest = (sav_balance * 2) / 100;
        sav_balance = sav_balance + interest;
    }
    void s_withdraw()
    {
        float withdraw;
        cout << "\nBalance :- " << sav_balance;
        cout << "\nEnter amount to be withdraw : ";
        cin >> withdraw;
        if (sav_balance > 500)
        {
            sav_balance = sav_balance - withdraw;
            cout << "\nBalance Amount After Withdraw: " << sav_balance;
        }
        else
        {

```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        cout << "\n Insufficient Balance";
    }
}
};

int main()
{
    current_account c1;
    saving_account s1;
    char type;
    cout << "\nEnter S for saving customer and C for current a/c customer : ";
    cin >> type;
    int choice;
    if (type == 's' || type == 'S')
    {
        s1.getAccountDetails();
        while (1)
        {
            cout << "\nChoose Your Choice" << endl;
            cout << "1)    Deposit" << endl;
            cout << "2)    Withdraw" << endl;
            cout << "3)    Display Balance" << endl;
            cout << "4)    Display with full Details" << endl;
            cout << "5)    Exit" << endl;
            cout << "Enter Your choice: ";
            cin >> choice;
            switch (choice)
            {
                case 1:
                    s1.s_deposit();
                    break;
                case 2:
                    s1.s_withdraw();
                    break;
                case 3:
                    s1.s_display();
                    break;
                case 4:
                    s1.displayDetails();
                    s1.s_display();
            }
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        break;
    case 5:
        goto end;
    default:
        cout << "\n\nEntered choice is invalid,\"TRY AGAIN\"";
    }
}
}
else if (type == 'c' || type == 'C')
{
    c1.getAccountDetails();
    while (1)
    {
        cout << "\nChoose Your Choice" << endl;
        cout << "1)    Deposit" << endl;
        cout << "2)    Withdraw" << endl;
        cout << "3)    Display Balance" << endl;
        cout << "4)    Display with full Details" << endl;
        cout << "5)    Exit" << endl;
        cout << "Enter Your choice: ";
        cin >> choice;
        switch (choice)
        {
            case 1:
                c1.c_deposit();
                break;
            case 2:
                c1.c_withdraw();
                break;
            case 3:
                c1.c_display();
                break;
            case 4:
                c1.displayDetails();
                c1.c_display();
                break;
            case 5:
                goto end;
            default:
                cout << "\n\nEntered choice is invalid,\"TRY AGAIN\"";
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        }  
    }  
}  
else  
{  
    cout << "\nInvalid Account Selection";  
}  
end:  
cout << "\nThank You for Banking with us..";  
return 0;  
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:

```
Enter S for saving customer and C for current a/c customer : c
```

```
Enter Customer Name : Shashwat  
Enter Account Number : 456972  
Enter Account Type : Savings
```

```
Choose Your Choice
```

- 1) Deposit
- 2) Withdraw
- 3) Display Balance
- 4) Display with full Details
- 5) Exit

```
Enter Your choice: 1
```

```
Enter amount to Deposit : 466710
```

```
Choose Your Choice
```

- 1) Deposit
- 2) Withdraw
- 3) Display Balance
- 4) Display with full Details
- 5) Exit

```
Enter Your choice: 4
```

```
Customer Name : Shashwat  
Account Number : 456972  
Account Type : Savings  
Balance :471710
```

```
Choose Your Choice
```

- 1) Deposit
- 2) Withdraw
- 3) Display Balance
- 4) Display with full Details
- 5) Exit

```
Enter Your choice: 5
```

```
Thank You for Banking with us.._
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 4:

Q 4.) An election is contested by five candidates. The candidates are numbered 1 to 5 and a voting is done by marking the candidate number in a ballot paper. Write a C++ program to read the ballot and count the votes cast for each candidate using an array variable count. In case, a number read is outside the range 1 to 5 the ballot should be considered as a 'spoilt ballot', and the program should also count the number of spoilt ballots.

CODE:-

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

void Menu()
{
    cout << "\n1 - first candidate"
         << "\n2 - second candidate"
         << "\n3 - third candidate"
         << "\n4 - fourth candidate"
         << "\n5 - fifth candidate"
         << "\n0 - exit program" << endl;
    cout << "Your choice: ";
}

int main()
{
    int n;
    int candidates[6] = {0, 0, 0, 0, 0, 0};
    do
    {
        Menu();
        cin >> n;
        switch (n)
        {
            case 1:
            {
                candidates[0]++;
                break;
            }
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
    }  
    case 2:  
    {  
        candidates[1]++;  
        break;  
    }  
    case 3:  
    {  
        candidates[2]++;  
        break;  
    }  
    case 4:  
    {  
        candidates[3]++;  
        break;  
    }  
    case 5:  
    {  
        candidates[4]++;  
        break;  
    }  
    default:  
    {  
        candidates[5]++;  
        break;  
    }  
    }  
} while (n != 0);  
cout << "\nResults of votes:";  
cout << "\nFirst candidate " << candidates[0]  
    << "\nSecond candidate " << candidates[1]  
    << "\nThird candidate " << candidates[2]  
    << "\nFourth candidate " << candidates[3]  
    << "\nFifth candidate " << candidates[4]  
    << "\nSpoilt ballots " << candidates[5];  
}
```


Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:

```
1 - first candidate
2 - second candidate
3 - third candidate
4 - fourth candidate
5 - fifth candidate
0 - exit program
Your choice: 2
```

```
1 - first candidate
2 - second candidate
3 - third candidate
4 - fourth candidate
5 - fifth candidate
0 - exit program
Your choice: 4
```

```
1 - first candidate
2 - second candidate
3 - third candidate
4 - fourth candidate
5 - fifth candidate
0 - exit program
Your choice: 2
```

```
1 - first candidate
2 - second candidate
3 - third candidate
4 - fourth candidate
5 - fifth candidate
0 - exit program
Your choice: 6
```

```
1 - first candidate
2 - second candidate
3 - third candidate
4 - fourth candidate
5 - fifth candidate
0 - exit program
Your choice: 0
```

```
Results of votes:
First candidate 0
Second candidate 2
Third candidate 0
Fourth candidate 1
Fifth candidate 0
Spoilt ballots 2
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 5:

Q 5.) Develop a program which will read a string and rewrite it in the alphabetical order. For example, the word STRING should be written as GINRST.

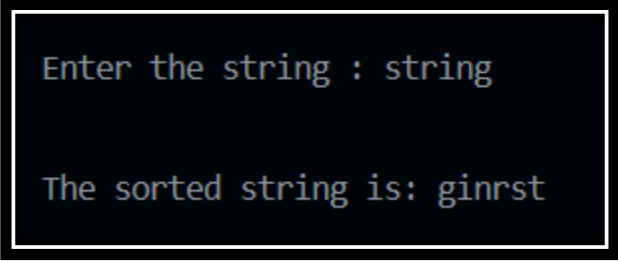
CODE:-

```
#include <iostream>
#include <string.h>
using namespace std;

int main()
{
    char string[100];
    cout << "\n\nEnter the string : ";
    cin >> string;
    char temp;
    int i, j;
    int n = strlen(string);
    for (i = 0; i < n - 1; i++)
    {
        for (j = i + 1; j < n; j++)
        {
            if (string[i] > string[j])
            {
                temp = string[i];
                string[i] = string[j];
                string[j] = temp;
            }
        }
    }
    cout << "\n\nThe sorted string is: " << string;
    cout<<"\n\n";
    return 0;
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:



```
Enter the string : string
```

```
The sorted string is: ginrst
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 6:

Q 6.) Create a class by name date with the member data day, month and year. Perform the following:

- Overload all relational operators <,<=,>,>=,==,!=
- Overload ++ operator to increment a date by one day
- Overload + to add given number of days to find the next date
- Provide the necessary function to use the statement like days=dt; where days is an int variable and dt is an object of date class. The statement is intended to assign the number of days elapsed in the current year of the date to the variable days. Note that this is a case of conversion from derived type to basic type.

CODE:-

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

class date
{
private:
    int day;
    int month;
    int year;
public:
    date();
    date(int dt, int mn, int yrs)
    {
        day = dt;
        month = mn;
        year = yrs;
    }
    void getdate(int dt, int mn, int yrs)
    {
        cout << "enter day month year";
        cin >> day >> month >> year;
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
void printdate()
{
    cout << "\n"
         << day << "-" << month << "-" << year << "\n";
}
void operator++();
bool operator<(date dt);
bool operator>(date dt);
bool operator!=(date dt);
bool operator==(date dt);
bool operator<=(date dt);
bool operator>=(date dt);
void operator+(int days);
operator int() const;
};

void date::operator++()
{
    ++day;
}

bool date::operator<(date dt)
{
    if ((year == dt.year) && (month == dt.month))
    {
        if (day < dt.day)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else if ((year == dt.year) && (month != dt.month))
    {
        if (month < dt.month)
        {
            return true;
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        else
        {
            return false;
        }
    }
    else
    {
        if (year < dt.year)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

bool date::operator>(date dt)
{
    if ((year == dt.year) && (month == dt.month))
    {
        if (day > dt.day)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else if ((year == dt.year) && (month != dt.month))
    {
        if (month > dt.month)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
    }  
}  
else  
{  
    if (year > dt.year)  
    {  
        return true;  
    }  
    else  
    {  
        return false;  
    }  
}  
}  
  
bool date::operator!=(date dt)  
{  
    if ((day != dt.day) || (month != dt.month) || (year != dt.year))  
    {  
        return true;  
    }  
    else  
    {  
        return false;  
    }  
}  
  
bool date::operator==(date dt)  
{  
    if ((day == dt.day) && (month == dt.month) && (year == dt.year))  
    {  
        return true;  
    }  
    else  
    {  
        return false;  
    }  
}  
  
bool date::operator<=(date dt)
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
{
    if ((year == dt.year) && (month == dt.month))
    {
        if (day <= dt.day)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else if ((year == dt.year) && (month != dt.month))
    {
        if (month <= dt.month)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else
    {
        if (year <= dt.year)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

bool date::operator>=(date dt)
{
    if ((year == dt.year) && (month == dt.month))
    {
```


Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        if (day >= dt.day)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else if ((year == dt.year) && (month != dt.month))
    {
        if (month >= dt.month)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else
    {
        if (year >= dt.year)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

void date::operator+(int days)
{
    int temp = day + days;
    if (month == 2)
    {
        if (temp > 28)
        {

```

Registration No: **21BCE11173**

Name : **SHASHWAT SHANDILYA**

Division: **CSE- CORE**

```
        month = month + 1;
        temp = temp % 31;
        day = day + temp;
    }
}
else if ((month % 2 != 0 || month == 8 || month == 10) && (month != 9 ||
month != 11))
{
    if (temp > 31)
    {
        month = month + 1;
        temp = temp % 31;
        day = temp;
    }
    else
    {
        day = day + temp;
    }
}
else
{
    if (temp > 30)
    {
        month = month + 1;
        temp = temp % 30;
        day = temp;
    }
    else
    {
        day = day + temp;
    }
}
}

date ::operator int() const
{
    int days, months;
    days = day;
    months = month;
    for (int i = 0; i < months; i++)
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
{
    if (i == 2)
        days += 28;
    else if (i == 4 || i == 6 || i == 9 || i == 11)
        days += 30;
    else
        days += 31;
}

return days;
}

int main()
{
    date da(8, 6, 2002);
    date db(9, 6, 2002);
    da.printdate();
    db.printdate();
    cout << "\nda<db  " << std::boolalpha << bool(da < db) << "\n";
    cout << "da>db  " << std::boolalpha << bool(da > db) << "\n";
    cout << "da==db " << std::boolalpha << bool(da == db) << "\n";
    cout << "da!=db " << std::boolalpha << bool(da != db) << "\n";
    cout << "da<=db " << std::boolalpha << bool(da <= db) << "\n";
    cout << "da>=db " << std::boolalpha << bool(da >= db) << "\n";
    cout << "\n\nadding 31 day \n";
    cout << "Old date ";
    da.printdate();
    da + 31;
    cout << "New date ";
    da.printdate();
    cout << "\nIncrementing date by 1 day ";
    ++da;
    da.printdate();

    int days;
    days = da;
    cout << "Total days elapsed in da :";
    cout << days << endl<< endl;
    return 0;
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:

```
8-6-2002
```

```
9-6-2002
```

```
da<db  true  
da>db  false  
da==db false  
da!=db true  
da<=db true  
da>=db false
```

```
adding 31 day  
Old date  
8-6-2002  
New date  
9-7-2002
```

```
Incrementing date by 1 day  
10-7-2002  
Total days elapsed in da :222
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 7:

Q 7.) Develop a program to sort a file consisting of books' details in the alphabetical order of author names.

The details of books include book_id, author_name, price, no_of_pages, publisher, year_of_publishing.

CODE:-

```
#include <iostream>
#include <vector>

using namespace std;

class Book
{
public:
    int book_id;
    char author_name[20];
    float book_price;
    int no_of_pages;
    char publisher[20];
    int publication_year;
    void getdata()
    {
        cout << endl;
        cout << "\nEnter the book id: ";
        cin >> book_id;
        cout << "\nEnter the author name: ";
        cin >> author_name;
        cout << "\nEnter the book price: ";
        cin >> book_price;
        cout << "\nEnter the number of pages: ";
        cin >> no_of_pages;
        cout << "\nEnter the publisher: ";
        cin >> publisher;
        cout << "\nEnter the publication year: ";
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        cin >> publication_year;
    }
    void putdata()
    {
        cout << endl;
        cout << "\nBook id: " << book_id << endl;
        cout << "\nAuthor name: " << author_name << endl;
        cout << "\nBook price: " << book_price << endl;
        cout << "\nNumber of pages: " << no_of_pages << endl;
        cout << "\nPublisher: " << publisher << endl;
        cout << "\nPublication year: " << publication_year << endl;
    }
};

int main()
{
    int n;
    cout << "\n\nEnter the number of books: ";
    cin >> n;
    vector<Book> v;
    for (int i = 0; i < n; i++)
    {
        Book b;
        b.getdata();
        v.push_back(b);
    }

    // alphabetic sort of author names
    // implementing individual character comparison bubble sort

    int indexA1, indexA2;

    for (int i = 0; i < n - 1; i++)
    {
        indexA1 = indexA2 = -1;
        for (int j = 0; j < n - 1; j++)
        {
            startComp:
                indexA1++;
                indexA2++;
        }
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
        char strCompA1 = v[j].author_name[indexA1];
        char strCompA2 = v[j + 1].author_name[indexA1];

        if (strCompA1 == strCompA2)
        {
            goto startComp;
        }
        else if (strCompA1 > strCompA2)
        {
            Book temp;
            temp = v[j];
            v[j] = v[j + 1];
            v[j + 1] = temp;
        }
    }
}

for (int i = 0; i < n; i++)
{
    v[i].putdata();
}
return 0;
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

OUTPUT:

```
Enter the number of books: 2

Enter the book id: 457

Enter the author name: Shashwat

Enter the book price: 4550

Enter the number of pages: 1200

Enter the publisher: XYZ

Enter the publication year: 2003

Enter the book id: 691

Enter the author name: Trisha

Enter the book price: 5000

Enter the number of pages: 690

Enter the publisher: Abc

Enter the publication year: 2005
```

```
Book id: 457

Author name: Shashwat

Book price: 4550

Number of pages: 1200

Publisher: XYZ

Publication year: 2003

Book id: 691

Author name: Trisha

Book price: 5000

Number of pages: 690

Publisher: Abc

Publication year: 2005
```


Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 8:

Q 8.) Design a class template by name Vector and perform the following:

- Find the smallest of the element in the Vector.
- Search for an element in the Vector.
- Find the average of the element in the array.

CODE:-

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

template <class T>
class Vector
{
    vector<T> vec{0, 0, 0, 0, 0};

public:
    void input()
    {
        cout << "\n\nEnter Element : ";
        for (int i = 0; i < 5; i++)
        {
            cin >> vec[i];
        }
    }

    void minele()
    {
        cout << "\nSmallest Element is : " << *min_element(vec.begin(),
vec.end()) << endl;
        return;
    }
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
void search()
{
    T x; T i;
    cout << "\nEnter element to find : ";
    cin >> x;

    for (i = 0; i < 5; i++)
    {
        if (vec.at(i) == x)
        {
            cout << "\nElement found at index : " << i << endl;
            return;
        }
    }

    cout << "\nElement not found" << endl
        << endl;
    return;
}

void average()
{
    int total = 0;
    for (auto i : vec)
    {
        total += i;
    }

    cout << "\nAverage : " << total / 5 << endl;
}

};

int main()
{
    cout << "\n\nInteger Type: " << endl;
    Vector<int> obj;
    obj.input();
    obj.minele();
    obj.search();
    obj.average();
}
```

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

```
cout << "\n\nDouble Type: " << endl;
Vector<double> obj1;
obj1.input();
obj1.minele();
obj.search();
obj.average();

return 0;
}
```

OUTPUT:

Integer Type:

Enter Element : 12 23 36 69 45

Smallest Element is : 12

Enter element to find : 69

Element found at index : 3

Average : 37

Double Type:

Enter Element : 2.5 3.1 4.6 7.2 9.1

Smallest Element is : 2.5

Enter element to find : 10.3

Element not found

Average : 37

Registration No: **21BCE11173**
Name : **SHASHWAT SHANDILYA**
Division: **CSE- CORE**

EXPERIMENT 9:

Q 9.) Design a generic function for finding the largest of three numbers.

CODE:-

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

template <typename T>

T myMax(T x, T y, T z)
{
    return (x > y) ? ((x > z) ? x : z) : ((y > z) ? y : z);
}

int main()
{
    int a, b, c;
    cout << "\n\nEnter the three numbers (space-separated): ";
    cin >> a >> b >> c;
    cout << myMax(a, b, c) << endl<<endl;

    return 0;
}
```

OUTPUT:

```
Enter the three numbers (space-separated): 45 69 12
69
```
