**Session 4 (unit-2): Dynamic memory allocation (new & delete) and Inline vs. Macros**

|  |  |
| --- | --- |
| Roll No. A016 | Name: Varun Khadayate |
| Program: B-Tech (CSBS) | Division: SY |
| Batch: 1 | Date of Submission: 20-09-2020 |

1. **Show how the Dynamic memory can be allocated and de-allocated to arrays in C++**

**ANS:**

**CODE:** #include<iostream>

using namespace std;

int main()

{

int \*x,\*y;

x = new int; //new = allocation

y = new int;

cout<<"Enter value of x::";

cin>>\*x;

cout<<"\nEnter value of y::";

cin>>\*y;

cout<<"\nEntered value for x::"<< \*x << endl;

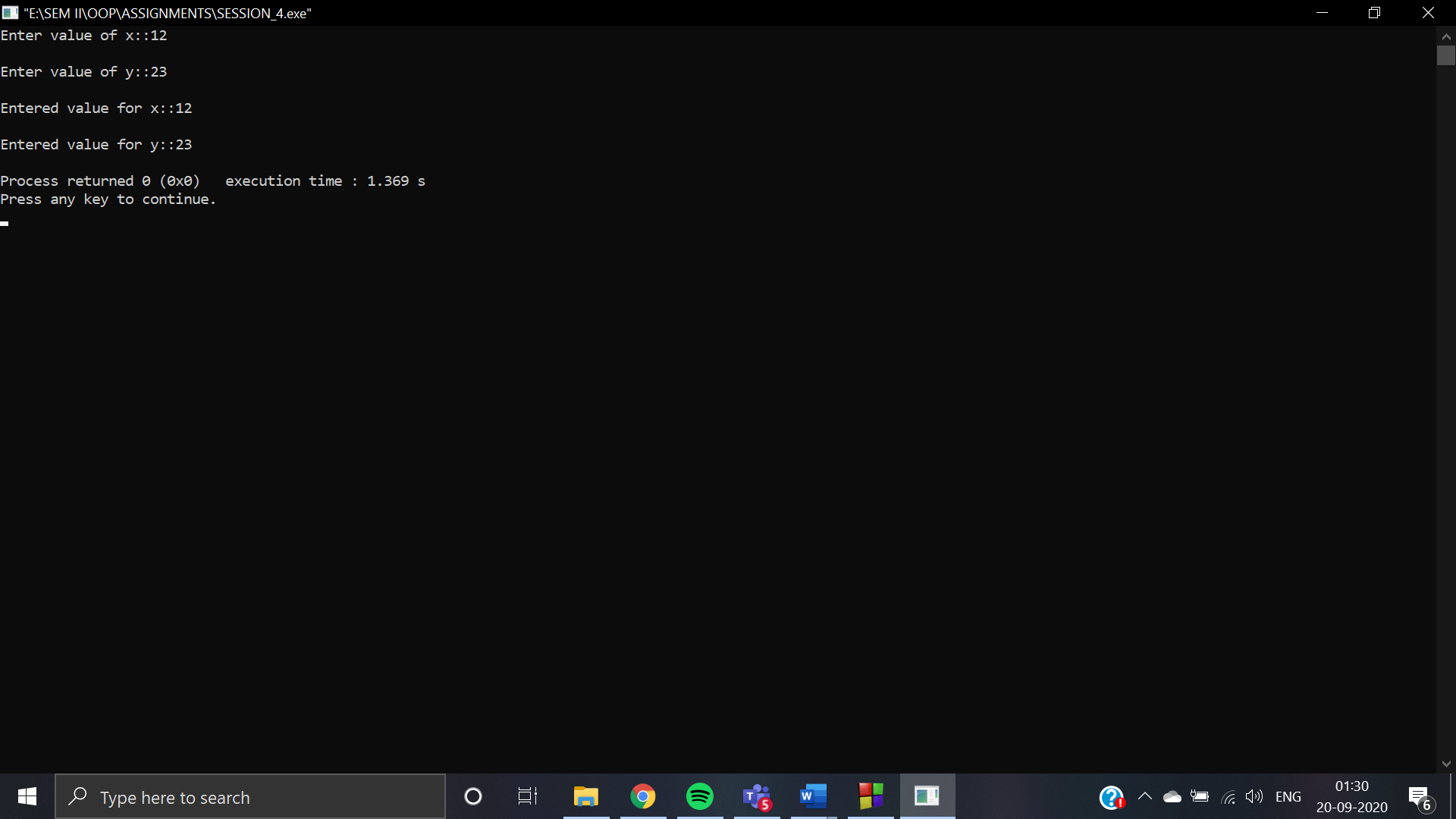
cout<<"\nEntered value for y::"<< \*y << endl;

delete x, y; //delete = deallocation

return 0;

}

**OUTPUT:**



1. **Write a macro to calculate simple interest from principal, rate of interest and time. Simple interest = (principal\*rate of interest\*time)/100.**

**ANS:**

**CODE:** #include<iostream>

using namespace std;

int main()

{

float principal\_amt, rate, simple\_interest;

int time;

cout<<"Enter the values of principal\_amt::";

cin>>principal\_amt;

cout<<"Enter the values of rate::";

cin>>rate;

cout<<"Enter the values of time::";

cin>>time;

simple\_interest = (principal\_amt \* rate \* time) / 100.0;

cout<<"Amount = Rs."<<principal\_amt<<"\n";

cout<<"Rate = Rs."<<rate<<"\n";

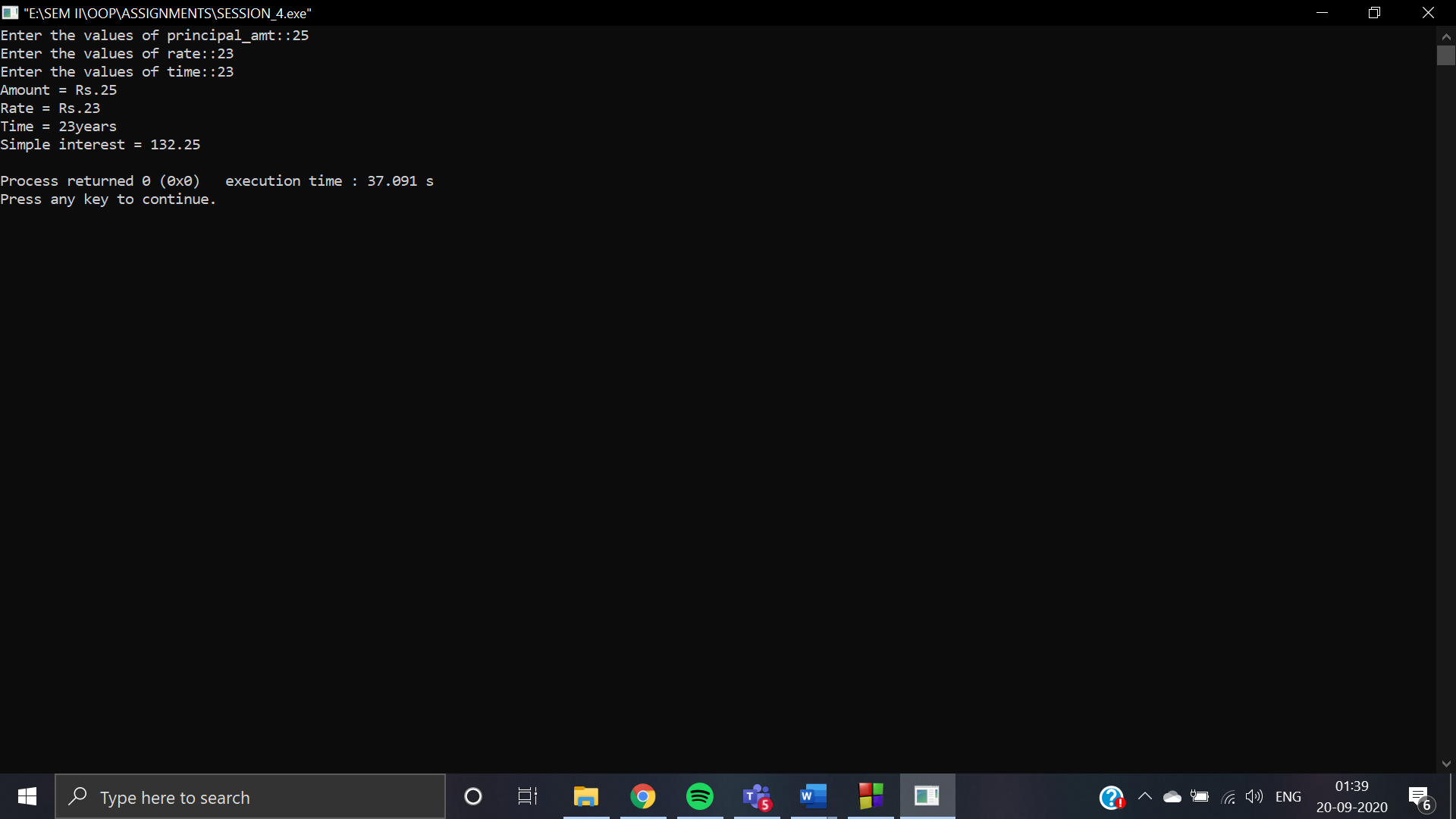
cout<<"Time = "<<time<<"years\n";

cout<<"Simple interest = "<<simple\_interest<<"\n";

return 0;

}

**OUTPUT:**



1. **WAP that can demonstrate the functionality of a calculator and it must include the following operations:  
     
   addition, subtraction, division, multiplication and getting two input values from the user.**

**You must perform implementation of all these operations using three cases:**

**Case 1: using normal functions**

**Case 2: using Inline functions**

**Case 3: using macros**

**Comment which will be the most suitable way of implementation in this problem and why?**

**ANS:**

* **Case 1: using normal functions**

**CODE:** #include<iostream>

using namespace std;

int main()

{

char opertor;

int x, y;

cout<<"Enter an operator (+, -, \*,): ";

cin>>opertor;

cout<<"Enter two operands: ";

cin>>x>>y;

switch (opertor)

{

case '+':

cout<<x<<"+"<<y<<"="<<x + y;

break;

case '-':

cout<<x<<"-"<<y<<"="<<x - y;

break;

case '\*':

cout<<x<<"\*"<<y<<"="<<x \* y;

break;

case '/':

cout<<x<<"/"<<y<<"="<<x / y;

break;

default:

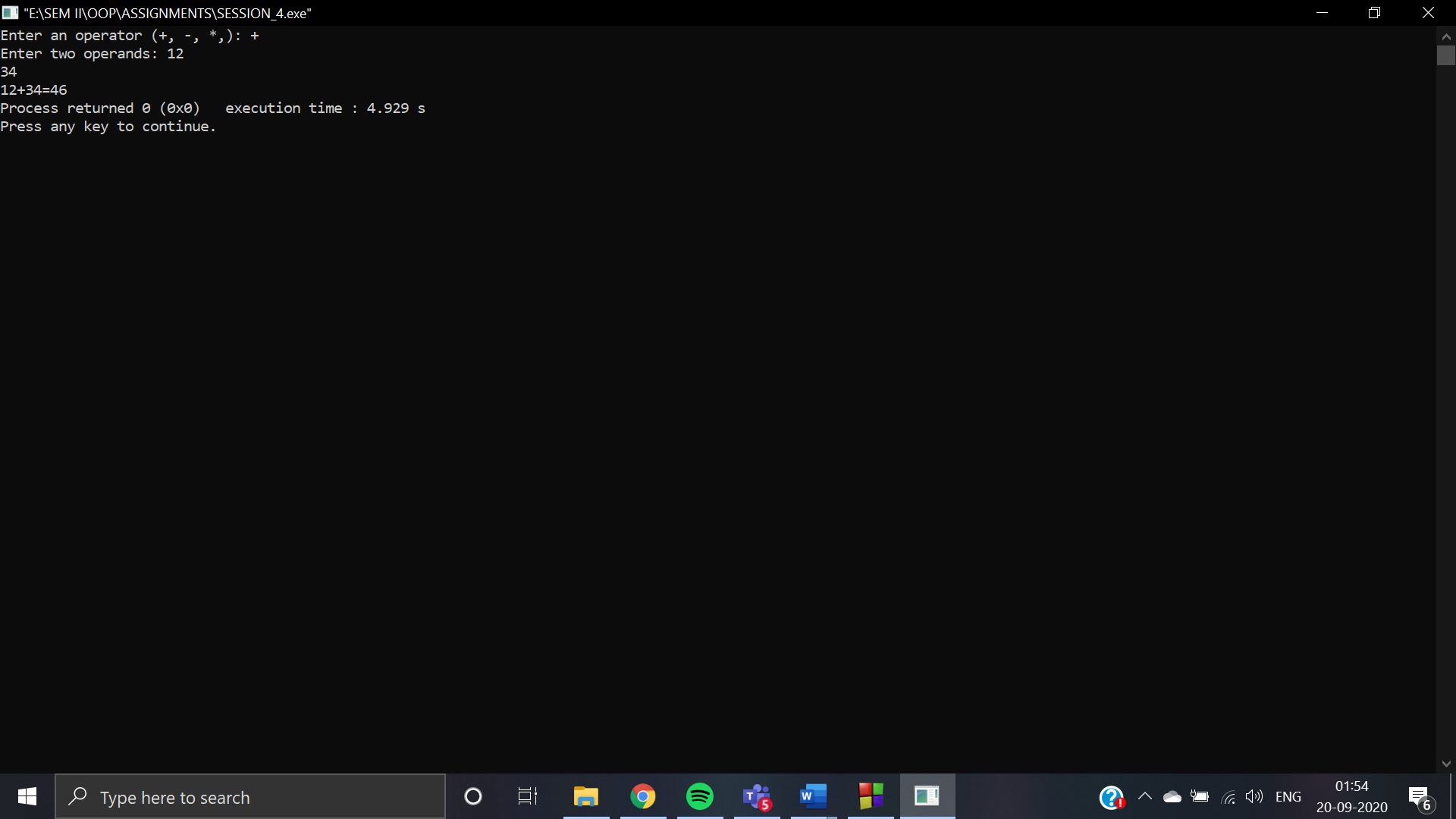
cout<<"Error! operator is not correct";

}

return 0;

}

**OUTPUT:**



* **Case 2: using Inline functions**

**CODE:** #include <iostream>

using namespace std;

class operation

{

int a,b,add,sub,mul;

float div;

public:

void get();

void sum();

void difference();

void product();

void division();

};

inline void operation :: get()

{

cout << "Enter first value:";

cin >> a;

cout << "Enter second value:";

cin >> b;

}

inline void operation :: sum()

{

add = a+b;

cout << "Addition of two numbers: " << a+b << "\n";

}

inline void operation :: difference()

{

sub = a-b;

cout << "Difference of two numbers: " << a-b << "\n";

}

inline void operation :: product()

{

mul = a\*b;

cout << "Product of two numbers: " << a\*b << "\n";

}

inline void operation ::division()

{

div=a/b;

cout<<"Division of two numbers: "<<a/b<<"\n" ;

}

int main()

{

operation s;

s.get();

s.sum();

s.difference();

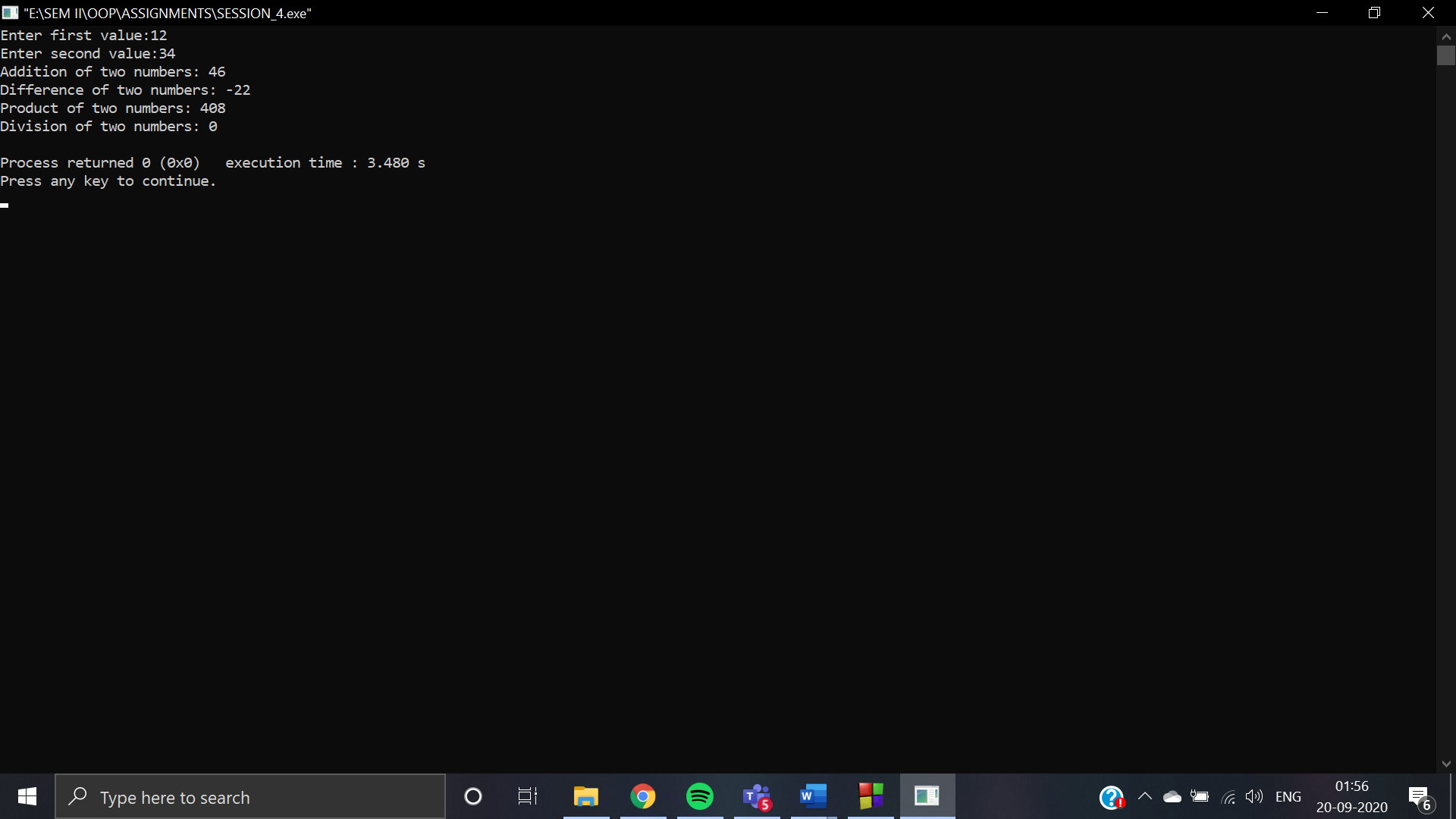
s.product();

s.division();

return 0;

}

**OUTPUT:**



* **Case 3: using macros**

**CODE:** #include<iostream>

#include<stdio.h>

#include<stdlib.h>

using namespace std;

#define ADD(X, Y) ( X + Y)

#define SUBTRACT(X, Y) (X - Y)

#define MULTIPLY(X, Y) (X \* Y)

#define DIVIDE(X, Y) (X/Y)

int main()

{

int a,b;

int addition,subtraction,mul,division;

addition = subtraction = mul = division = 0;

cout<<"Enter two numbers:"<<"\n";

cin>>a;

cin>>b;

addition = ADD(a, b);

subtraction = SUBTRACT(a, b);

mul = MULTIPLY(a, b);

division = DIVIDE(a, b);

cout<<"\n";

cout<<"ANSWERS"<<"\n";

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<"\n";

cout<<"Addition ="<<addition<<"\n";

cout<<"Subtraction ="<<subtraction<<"\n";

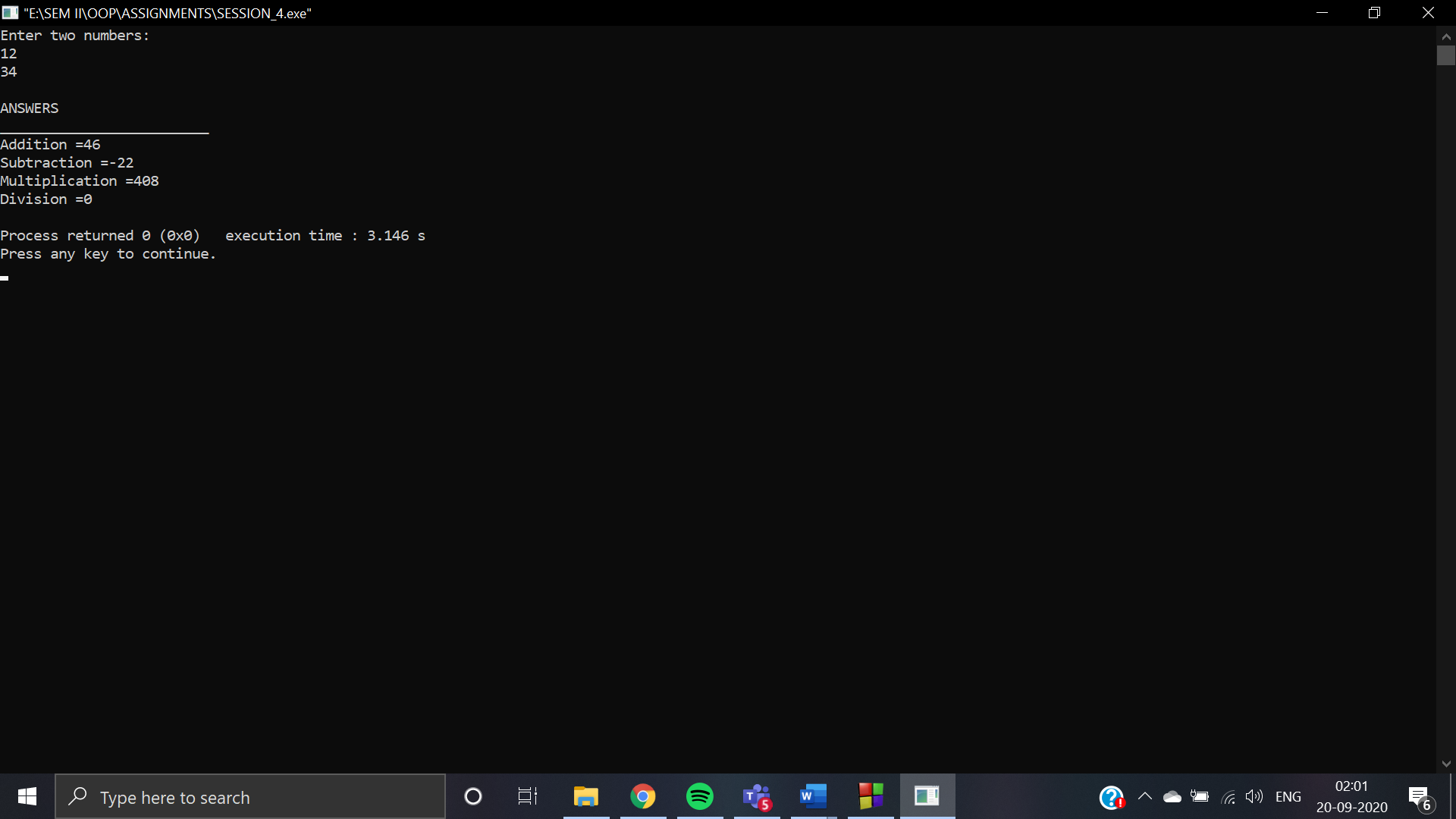
cout<<"Multiplication ="<<mul<<"\n";

cout<<"Division ="<<division<<"\n";

return 0;

}

**OUTPUT:**



**Macro is more suitable according to me because macros are expanded when the program is processed by the pre-processor which makes easy to code.**