/\*Task2:

Write a recursive function to print a Fibonacci series up to N numbers

Enter the number of elements: 10

Fibonacci Series: 0 1 1 2 3 5 8 13 21 34

Note: Use minimum lines of code as possible to get max marks.\*/

#include <iostream>

using namespace std;

void fibonacci(int n, int& prev, int& curr, int count)

{

if (count == n)

{

return;

}

else

{

cout << prev << " ";

static int next;

next = prev + curr;

prev = curr;

curr = next;

fibonacci(n, prev, curr, count + 1);

}

}

int main()

{

int num;

int prev = 0, curr = 1, count=0;

cout << "Enter the number of elements: ";

cin >> num;

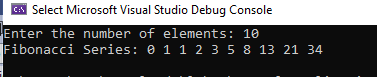
cout << "Fibonacci Series: ";

fibonacci(num, prev, curr, count);

cout << endl;

return 0;

}



/\*Task 3:

The greatest common divisor of integers x and y is the largest integer that evenly divides both x and y. Write

a recursive function gcd that returns the greatest common divisor of x and y, defined recursively as follows:

If y is equal to 0, then gcd(x, y) is x; otherwise, gcd(x, y) is gcd(y, x % y), where % is the modulus operator.

[Note: For this algorithm, x must be larger than y.]\*/

#include <iostream>

using namespace std;

int gcd(int x, int y)

{

if (y == 0)

{

return x;

}

else

{

return gcd(y, x % y);

}

}

int main()

{

int x, y;

cout << "Enter two integers: ";

cin >> x >> y;

if (x < y)

{

swap(x, y);

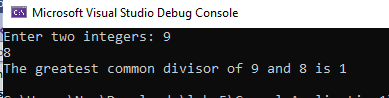
}

int result = gcd(x, y);

cout << "The greatest common divisor of " << x << " and " << y << " is " << result << endl;

return 0;

}



/\*Task 4:

Can main be called recursively on your system? Write a program containing a function main. Include static

local variable count and initialize it to 1. Postincrement and print the value of count each time main is called.

Compile your program. What happens?\*/

#include <iostream>

using namespace std;

int main()

{

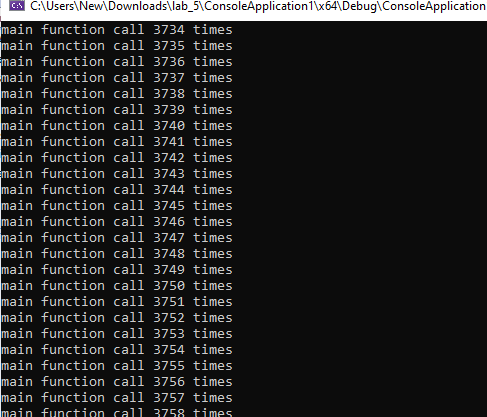
static int count = 1;

cout << "main function call " << count++ << " times" << endl;

main();

return 0;

}



/\*Task 5:

Find min and max value from array using recursion.\*/

#include <iostream>

using namespace std;

void findMinMax(int arr[], int n, int& min, int& max)

{

if (n == 1)

{

min = arr[0];

max = arr[0];

return;

}

int min2, max2;

findMinMax(arr, n - 1, min, max);

if (arr[n - 1] < min)

{

min = arr[n - 1];

}

if (arr[n - 1] > max)

{

max = arr[n - 1];

}

}

int main()

{

int arr[] = { 5, 2, 7, 3, 9, 1 };

int n = sizeof(arr) / sizeof(arr[0]);

int min, max;

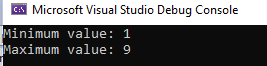
findMinMax(arr, n, min, max);

cout << "Minimum value: " << min << endl;

cout << "Maximum value: " << max << endl;

return 0;

}



/\*Task 6:

Create a Union that calculates the salaries of visiting or regular employees of university. Visiting employees

get 3000/hour with addition of 1000 extra bonus, while regular employees just get their salary. Your program

should first determine the type of employee then calculates and display its salary.\*/

#include <iostream>

#include <string>

using namespace std;

union Employee

{

double visitingSalary;

double regularSalary;

};

int main()

{

Employee emp;

string employeeType;

double hoursWorked;

cout << "Enter employee type (visiting or regular): ";

cin >> employeeType;

cout << "Enter hours worked: ";

cin >> hoursWorked;

if (employeeType == "visiting")

{

emp.visitingSalary = hoursWorked \* 3000 + 1000;

cout << "Visiting employee salary: " << emp.visitingSalary << endl;

}

else if (employeeType == "regular")

{

emp.regularSalary = hoursWorked \* 1000;

cout << "Regular employee salary: " << emp.regularSalary << endl;

}

else

{

cout << "Invalid employee type" << endl;

}

return 0;

}

