**Table of Contents**

|  |  |  |
| --- | --- | --- |
| S.No | Program | Remarks |
| 1 | Program to input two numbers and display their sum |  |
| 2 | Program to input a number and find prime or not |  |
| 3 | Program to find a number is palindrome or not |  |
| 4 | Program to input 10 numbers and find Min,Max, & Sum |  |
| 5 | Program to find demonstrate method overloading & overriding |  |
| 6 | Program to show Interface in C# |  |
| 7 | Program to perform multiple inheritance using interface |  |
| 8 | Structure to find the Simple Interest |  |
| 9 | Program to add & subtract using delegate |  |
| 10 | Finding the sum and reverse of digits using delegate |  |
| 11 | Implementation of Seal Class in C# |  |
| 12 | Program of stack data structure with stack operation |  |
| 13 | Program of queue data structure with queue operation |  |
| 14 | Program of performing dictionary operation |  |
| 15 | Program to demonstrate use of Hashset collection concept |  |
| 16 | Program to create a file and check if exists or not and if exists display the content of the file |  |
| 17 | Creating a directory and copy the file to the directory renaming it. |  |

**1. Write a program in console application to input two  
number and display their sum.**

using System;

using static System.Console;

namespace numbers\_sum

{

class Program

{

static void Main(string[] args)

{

WriteLine("Sum of Two Numbers ");

Write("Enter first Number ");

int a = Convert.ToInt32(ReadLine());

Write("Enter second Number ");

int b = Convert.ToInt32(ReadLine());

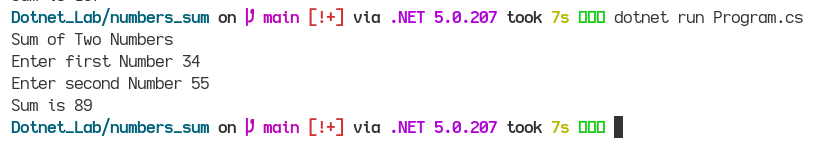
int sum = a + b;

WriteLine($"Sum is {sum}");

}

}

}

**OUTPUT:**

**2. Write a program to input a number and display it is prime  
number or not.**

using System;

using static System.Console;

namespace prime\_or\_not

{

class Program

{

static void Main(string[] args)

{

WriteLine("Checking Whether Prime or not");

Write("Enter a number ");

int number = Convert.ToInt32(ReadLine());

bool prime = true;

int i;

if (number == 0 || number == 1){

prime = false;

}

else{

for(i = 2; i <= (number/2); i++){

if (number % i == 0){

prime = false;

}

}

}

if (prime){

WriteLine($"{number} is prime");

}

else{

WriteLine($"{number} is not prime");

}

}

}

}

**OUTPUT:**



**3. Write a program to input a number and display it is  
palindrome or not.**

using System;

using static System.Console;

namespace palindrome\_check

{

class Program

{

static void Main(string[] args)

{

WriteLine("Checking The number is Palindrome or Not");

Write("Enter a Number ");

int number = Convert.ToInt32(ReadLine());

int r, sum = 0;

int temp = number;

while (number > 0){

r = number % 10;

sum = (sum \* 10) + r;

number = number / 10;

}

if (temp == sum){

WriteLine("Palindrome");

}

else{

WriteLine("Not Palindrome");

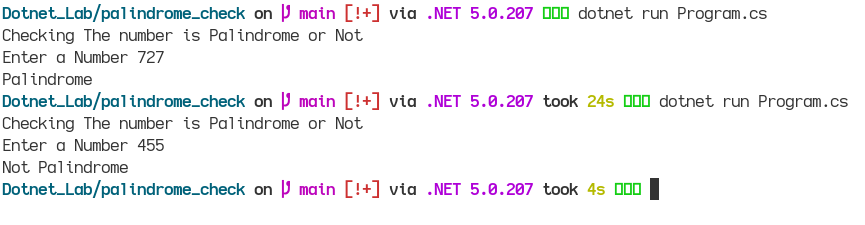
}

}

}

}

**OUTPUT:**



**4. Write a program to input 10 numbers and display in  
ascending order by using Array function. And also display Min, Max value and sum of Array.**

using System;

using static System.Console;

namespace array\_numbers

{

class Program

{

static void Main(string[] args)

{

var number = new int[10];

int i;

int sum = 0;

WriteLine("Enter 10 numbers");

for (i=0; i < 10; i++){

number[i] = Convert.ToInt32(ReadLine());

}

Array.Sort(number);

WriteLine("Arranging in ascending order");

foreach (int nums in number)

{

Write(nums+", ");

}

int max = number[0];

int min = number[0];

foreach(int nums in number){

sum = sum + nums;

if (nums > max){

max = nums;

}

else

{

min = nums;

}

}

WriteLine("\nThe Max value is "+max);

WriteLine("The Min number is "+min);

WriteLine("Sum of array is "+sum);

}

}

}

**5. Write a program to demonstrate constructor, method**

**overriding and method overloading.**

**Overloading:**

using System;

using static System.Console;

namespace constructorEg

{

class Car

{

string CarModel;

int CarYear;

public Car(int Year, string Model){

CarModel = Model;

CarYear = Year;

}

// Showing Overloading

public Car(string Model, int Year){

CarModel = Model;

CarYear = Year;

}

static void Main(string[] args)

{

Car obj = new Car(1996,"Mustang");

WriteLine("Car Model:"+obj.CarModel);

WriteLine("Car Year:"+obj.CarYear);

WriteLine("\nOverloading Example");

Car newCar = new Car("Impala", 1967); //Invoking Overloading method

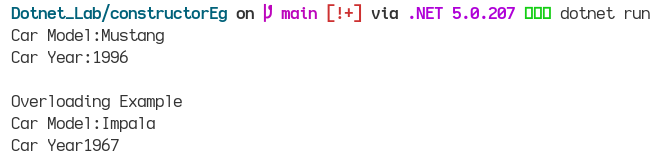
WriteLine("Car Model:"+newCar.CarModel);

WriteLine("Car Year"+newCar.CarYear);

}

}

}



**Overriding:**

using System;

namespace method\_Overriding

{

class Animal

{

public virtual void Eat(){

Console.WriteLine("Animal Eats Foods");

}

}

class Dog:Animal{

public override void Eat(){

Console.WriteLine("Dog Eats Foods");

}

public static void Main(string[] args){

Animal myanimal = new Animal();

myanimal.Eat();

Animal mydog = new Dog();

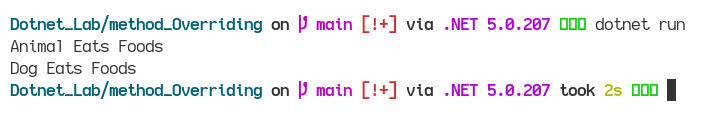
mydog.Eat();

}

}

}

**OUTPUT:**



**6. Write a program to demonstrate Interface and Also show  
program of multiple inheritance by using interface.**

using System;

namespace InterfaceEg

{

interface Iinterface1{

void method1();

void method2();

}

class Myclass1{

public void M1(){

Console.WriteLine("M1 from Class 1");

}

}

class Myclass2:Myclass1,Iinterface1{

public void method1(){

Console.WriteLine("Method 1");

}

public void method2(){

Console.WriteLine("Method 2");

}

}

class Program

{

static void Main(string[] args)

{

Myclass2 obj = new Myclass2();

obj.method1();

obj.method2();

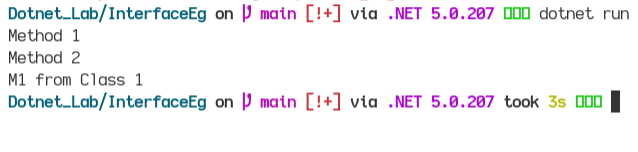
obj.M1();

}

}

}

**OUTPUT:**



**7. WAP to demonstrate multiple inheritance in .Net Framework**

using System;

using static System.Console;

using System.Collections;

namespace Interface\_multi\_Inheritance

{

interface Iinterface1{

void language();

}

class Program1:Iinterface1

{

public void language(){

ArrayList mylist = new ArrayList();

//adding elements in mylist array

mylist.Add("C");

mylist.Add("C#");

mylist.Add("Python");

mylist.Add("Java");

WriteLine("\n\*\*\*\*\*\*\*Language You need to Know :\*\*\*\*\*\*\*\*");

foreach (var elements in mylist)

{

WriteLine(elements);

}

}

}

interface Iinterface2{

void courses();

}

class Program2:Iinterface2{

public void courses(){

ArrayList mylist = new ArrayList();

mylist.Add("C fundamentals");

mylist.Add("C# classes");

mylist.Add("Python for data science");

mylist.Add("Java for applications");

WriteLine("\n\*\*\*\*\*\*\*Courses provided by Programming languages\*\*\*\*\*\* ");

foreach (var elements in mylist)

{

WriteLine(elements);

}

}

}

class Program: Iinterface1, Iinterface2{

Program1 obj1 = new Program1();

Program2 obj2 = new Program2();

public void language(){

obj1.language();

}

public void courses(){

obj2.courses();

}

}

class main\_Class{

static void Main(string[] args){

Program obj = new Program();

obj.language();

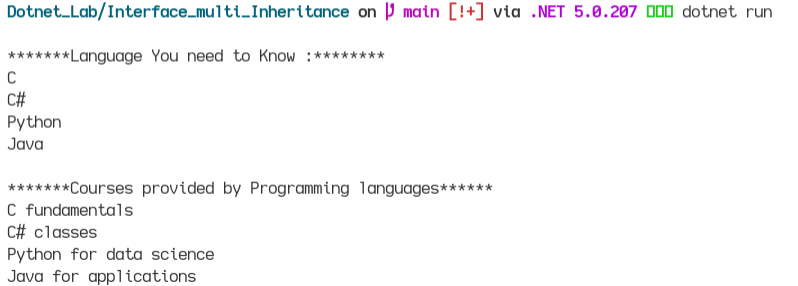
obj.courses();

}

}

}

**OUTPUT:**



**8. Write a program to create Structure type which store Principal,  
rate , time and display simple interest. Initialized the variable to default  
value with constructor.**

using System;

using static System.Console;

namespace structure\_with\_constructor

{

struct SimpleInterest{

public int principal;

public double rate;

public int time;

//Parameterized Constructor

public void getSI(int p, int t, double r){

double si = (p\*t\*r)/100;

WriteLine("Simple Interest is "+si);

}

}

class Program

{

static void Main(string[] args)

{

SimpleInterest obj = new SimpleInterest();

obj.principal = 100;

obj.time = 2;

obj.rate = 2.5;

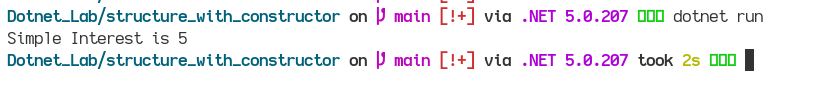
obj.getSI(obj.principal,obj.time,obj.rate);

}

}

}

**OUTPUT:**



**9. WAP to add, subtract of two number using delegate  
concept.**

using System;

using static System.Console;

namespace delegate\_sum\_diff

{

public delegate void myDelegate(int a , int b);

class Program

{

public void Sum(int x , int y){

WriteLine("Sum is "+ (x + y));

}

public void Diff(int x , int y){

WriteLine("Difference is "+(x-y));

}

static void Main(string[] args)

{

Program obj = new Program();

myDelegate sum\_diff = new myDelegate(obj.Sum);

sum\_diff += new myDelegate(obj.Diff);

WriteLine("Enter Two numbers\n");

Write("Enter first Number ");

int a = Convert.ToInt32(ReadLine());

Write("\nEnter second Number ");

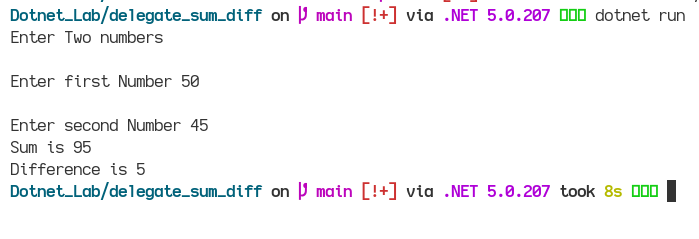
int b = Convert.ToInt32(ReadLine());

sum\_diff(a , b);

}

}

}



**10. WAP to find sum of digits and its reverse using delegate  
method.**

using System;

using static System.Console;

namespace digits\_delegates

{

// Program to Reverse and Sum the given digits using Delegate

public delegate void sumDelegates(int n);

class Program

{

public void reverseDigits(int digits\_number){

int rev = 0;

int sum = 0;

while (digits\_number > 0)

{

int i = digits\_number % 10;

rev = rev \* 10 + i;

sum = i + sum;

digits\_number= digits\_number / 10;

}

WriteLine("Reverse Digit is "+rev);

WriteLine("Sum is "+sum);

}

static void Main(string[] args)

{

Program obj = new Program();

sumDelegates digits = obj.reverseDigits;

WriteLine("Enter a number");

int num = Convert.ToInt32(ReadLine());

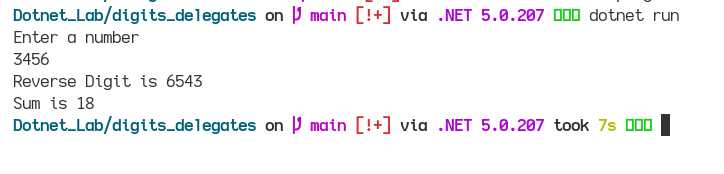
digits.Invoke(num);

}

}

}

**OUTPUT:**

****

**12. Create a seal class to demonstrate.**

using System;

using static System.Console;

namespace sealed\_demo

{

//Sealed identifier is used for sealed class

sealed class sealedDemo{

public void add(int x , int y){

WriteLine("Sum is "+(x+y));

}

}

class Program

{

static void Main(string[] args)

{

sealedDemo obj = new sealedDemo();

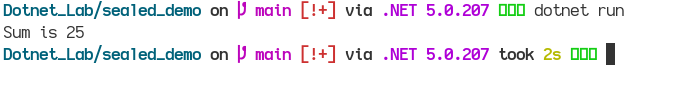
obj.add(10, 15);

}

}

}

**OUTPUT:**



**14. Crate a stack data structure and demonstrate different  
type of stack operation.**

using System;

using System.Collections;

using static System.Console;

using System.Threading;//For sleep execution

namespace stack\_operation

{

class Program

{

static void Main(string[] args)

{

Stack new\_stack = new Stack();

new\_stack.Push("Hello");

new\_stack.Push("Hi");

new\_stack.Push(33);

WriteLine("Elements after push operation in Stack");

foreach (var element in new\_stack)

{

Console.WriteLine(element);

}

Thread.Sleep(2000);

WriteLine("To check the string present in the stack ");

Thread.Sleep(2000);

WriteLine("\nEnter the string;(case sensitive)");

string message = ReadLine();

Thread.Sleep(2000);

if (new\_stack.Contains(message) == true)

{

WriteLine($"\*\*\*\*\*\*\*\*{message} is found\*\*\*\*\*\*\*");

}

else

{

WriteLine($"\*\*\*\*\*\*\*\*\*{message} cannot be found\*\*\*\*");

}

Thread.Sleep(2000);

WriteLine("\nNow after the pop() operation we have");

Thread.Sleep(2000);

new\_stack.Pop();

foreach (var element in new\_stack)

{

Console.WriteLine(element);

}

Thread.Sleep(2000);

WriteLine("\nElement on the top of the stack is ");

Thread.Sleep(3000);

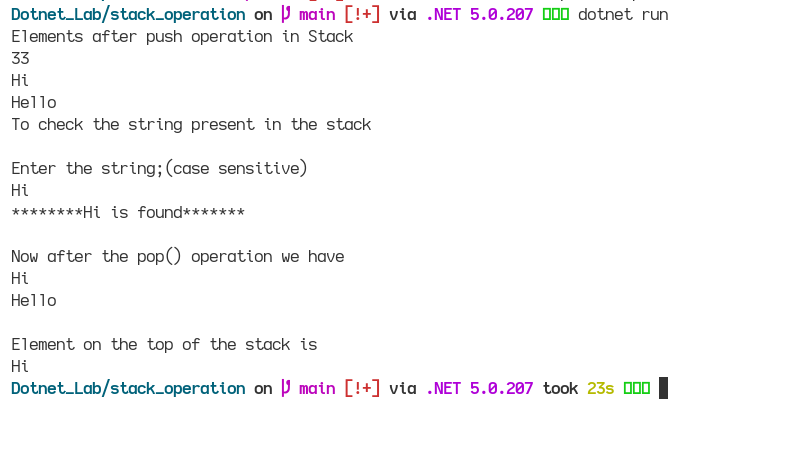
WriteLine(new\_stack.Peek());

}

}

}

**OUTPUT:**



**15. Create a Queue data structure and demonstrate different  
type of Queue operation.**

using System;

using System.Collections;

using static System.Console;

using System.Threading;

namespace queue\_operation

{

class Program

{

static void Main(string[] args)

{

Queue obj = new Queue();

obj.Enqueue("Hello");

obj.Enqueue("There");

obj.Enqueue("C#");

WriteLine("Objects in Queue are\n");

Thread.Sleep(2000);

foreach (var item in obj)

{

WriteLine(item);

}

WriteLine();

Thread.Sleep(2000);

WriteLine("Total Number of elements in queue is -->"+obj.Count);

Thread.Sleep(2000);

WriteLine();

WriteLine("To check the elements present in queue");

Thread.Sleep(2000);

WriteLine("Enter the string");

string message = ReadLine();

Thread.Sleep(2000);

if (obj.Contains(message) == true){

WriteLine($"\n{message} element found \n");

}

else{

WriteLine($"{message} element not found\n");

}

Thread.Sleep(2000);

WriteLine("Performing Dequeue operation");

obj.Dequeue();

WriteLine("After dequeue() operation, Elements are ");

WriteLine();

Thread.Sleep(2000);

foreach (var item in obj)

{

WriteLine(item);

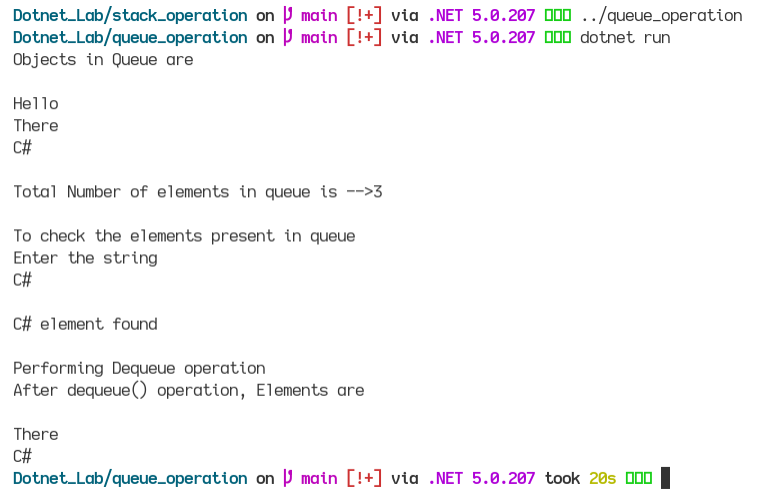
}

}

}

}

**OUTPUT:**



**16. Write a program to collect id and name of student in  
dictionary and find information about id 1 student and display it.**

using System;

using System.Collections.Generic;

using static System.Console;

namespace student\_dictionary

{

class Program

{

static void Main(string[] args)

{

Dictionary<int,string> dict\_obj = new Dictionary<int, string>();

dict\_obj.Add(2,"Ram");

dict\_obj.Add(3,"Hari");

dict\_obj.Add(1,"Rawan");

foreach (KeyValuePair<int,string> item in dict\_obj)

{

WriteLine($"Key:{item.Key} Value:{item.Value}");

}

WriteLine();

//Now finding the info of student with id 1 and display it

WriteLine("Display the student with id 1\n");

if(dict\_obj.ContainsKey(1) == true){

WriteLine($"Name of student with id 1 is '{dict\_obj[1]}'");

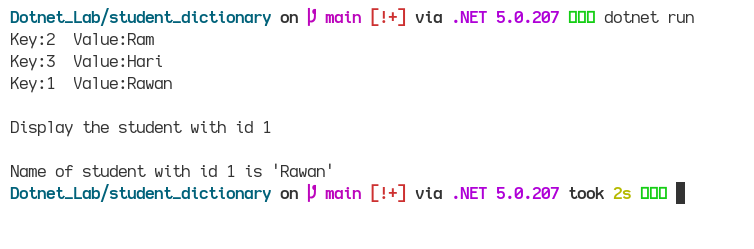
}

}

}

}

**OUTPUT:**



**17. Write a program to demonstrate use of Hashset collection  
concept.**

using System;

using System.Collections.Generic;

using static System.Console;

namespace student\_dictionary

{

class Program

{

static void Main(string[] args)

{

Dictionary<int,string> dict\_obj = new Dictionary<int, string>();

dict\_obj.Add(2,"Ram");

dict\_obj.Add(3,"Hari");

dict\_obj.Add(1,"Rawan");

foreach (KeyValuePair<int,string> item in dict\_obj)

{

WriteLine($"Key:{item.Key} Value:{item.Value}");

}

WriteLine();

//Now finding the info of student with id 1 and display it

WriteLine("Display the student with id 1\n");

if(dict\_obj.ContainsKey(1) == true){

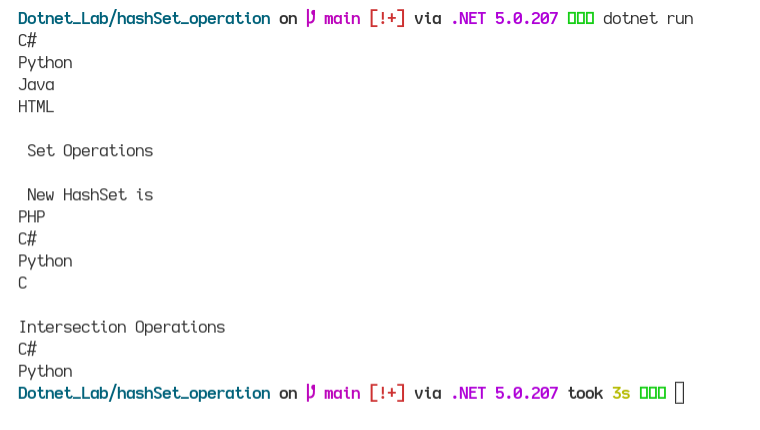
WriteLine($"Name of student with id 1 is '{dict\_obj[1]}'");

}

}

}

}



**18 & 19. Write a program to input a paragraph from a user and save  
in a file.**

using System;

using System.IO;

using static System.Console;

using System.Threading;

namespace write\_text\_file

{

class Program

{

static void Main(string[] args)

{

string file = "/home/tester13/Documents/Dotnet\_Lab/newfile"; //This is for my linux environment

WriteLine("Enter a sentence or a paragraph");

string message = ReadLine();

File.WriteAllText(file,message);

WriteLine();

WriteLine("Checking to read the file that exists or not");

if (File.Exists(file))

{

Thread.Sleep(2000);

WriteLine("File Exists");

Thread.Sleep(2000);

WriteLine();

WriteLine("The text of the file is");

Thread.Sleep(2000);

WriteLine(File.ReadAllText(file));

}

else

{

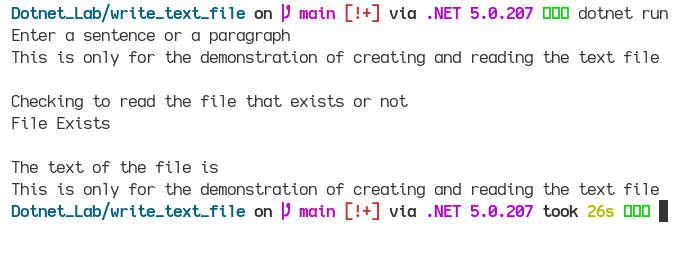
WriteLine("There is no such file");

}

}

}

}



**20. Create three folder A,B and C and copy the file you  
created in QNo. 18 to A folder by renaming it.**

using System;

using static System.Console;

using System.IO;

namespace create\_folder

{

class Program

{

static void Main(string[] args)

{

string folderPathA = "/home/tester13/Documents/Dotnet\_Lab/FolderA";

string folderPathB = "/home/tester13/Documents/Dotnet\_Lab/FolderB";

string folderPathC = "/home/tester13/Documents/Dotnet\_Lab/FolderC";

//Creating a directory

Directory.CreateDirectory(folderPathA);

Directory.CreateDirectory(folderPathB);

Directory.CreateDirectory(folderPathC);

string filename = "newfile";

string sourcePath = "/home/tester13/Documents/Dotnet\_Lab/";

string destfile = Path.Combine(folderPathA,filename);

string sourcefile = Path.Combine(sourcePath,filename);

File.Copy(sourcefile,destfile,true);

// To rename the file

File.Move($"{folderPathA}/newfile",$"{folderPathA}/ChangedFile");

}

}

}