|  |
| --- |
| **Day-10 evening assignment**  **By**  **Bhanu Rama Krishna Prakash Jakkamsetti**  **04/02/2022** |

|  |
| --- |
| 1.Research and try to understand what is Abstraction |
| Abstraction is an important part of object oriented programming. It means that only the required information is visible to the user and the rest of the information is hidden. Abstraction can be implemented using abstract classes in c#. Abstract classes are base classes with partial implementation. And in the abstract methods only have method declaration. |

|  |
| --- |
| 2.Write the 2 main uses of Abstract class by using the example discussed in the class. |
| Two uses of Abstract class : |
| 1. Reusability 2. The derived class must implement the Abstract methods. |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day10\_evng\_project1  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* Purpose: creating abstract class and methods  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  abstract class Salary  {  /// <summary>  /// creating pf  /// </summary>  /// <param name="basic"></param>  /// <returns>pf</returns>  public int GetPF(int basic)  {  return 12 \* basic / 100;  }  /// <summary>  /// crating hra  /// </summary>  /// <param name="basic"></param>  /// <returns>hra</returns>  public int GetHRA(int basic)  {  return 40 \* basic / 100;  }  /// <summary>  /// creating abstract CA  /// </summary>  /// <returns></returns>  public abstract int GetCA();  /// <summary>  /// creating abstract SA  /// </summary>  /// <returns></returns>  public abstract int GetSE();    }  class NB : Salary  {  /// <summary>  /// implementing abstract getca  /// </summary>  /// <returns>ca</returns>  public override int GetCA()  {  return 2000;  }  /// <summary>  /// implementing abstract getsa  /// </summary>  /// <returns>sa</returns>  public override int GetSE()  {  return 1000;  }  }  class Google : Salary  {  /// <summary>  /// implementing abstract getca  /// </summary>  /// <returns>ca</returns>  public override int GetCA()  {  return 4000;  }  /// <summary>  /// implementing abstract getsa  /// </summary>  /// <returns>sa</returns>  public override int GetSE()  {  return 3000;  }  }  class IBM : Salary  {  /// <summary>  /// implementing abstract getca  /// </summary>  /// <returns>ca</returns>  public override int GetCA()  {  return 6000;  }  /// <summary>  /// implementing abstract getsa  /// </summary>  /// <returns>sa</returns>  public override int GetSE()  {  return 5000;  }  }  class FB : Salary  {  /// <summary>  /// implementing abstract getca  /// </summary>  /// <returns>ca</returns>  public override int GetCA()  {  return 8000;  }  /// <summary>  /// implementing abstract getsa  /// </summary>  /// <returns>sa</returns>  public override int GetSE()  {  return 7000;  }  }  internal class Program  {  static void Main(string[] args)  {  Console.WriteLine("procesess success");  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 3.Create one more example of your choice to demonstrate abstract class |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day10\_evng\_project2  {/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author:Bhanu Rama Krishna Prakash Jakkamsetti  \* Purpose:creating abstract class and methods for Payment  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  abstract class PaymentApp  {  /// <summary>  /// creating pay  /// </summary>  /// <param name="amount"></param>  /// <returns>amount</returns>  public int Pay(int amount)  {  return amount;  }  /// <summary>  /// ctrating receive  /// </summary>  /// <param name="amount1"></param>  /// <returns>amount1</returns>  public int Receive(int amount1)  {  return amount1;  }  /// <summary>  /// declaring commision  /// </summary>  /// <param name="amt"></param>  /// <returns></returns>  public abstract int Commission(int amt);  }  class Phonepay : PaymentApp  {  /// <summary>  /// implemention abstract method  /// </summary>  /// <param name="amt"></param>  /// <returns>commision</returns>  public override int Commission(int amt)  {  return 2 \* amt / 100;  }  }  class Paytm : PaymentApp  {  /// <summary>  /// implemention abstract method  /// </summary>  /// <param name="amt"></param>  /// <returns>commision</returns>  public override int Commission(int amt)  {  return 3 \* amt / 100;  }  }  class Gpay : PaymentApp  {  /// <summary>  /// implemention abstract method  /// </summary>  /// <param name="amt"></param>  /// <returns>commision</returns>  public override int Commission(int amt)  {  return 4 \* amt / 100;  }  }  class WhatsappPay : PaymentApp  {  /// <summary>  /// implemention abstract method  /// </summary>  /// <param name="amt"></param>  /// <returns>commision</returns>  public override int Commission(int amt)  {  return 5 \* amt / 100;  }  }  internal class Program  {  static void Main(string[] args)  {  Console.WriteLine("payment success");  Console.ReadLine();  }  }  } |
| Output: |
|  |