

Trường: ĐH CNTP TP.HCM Khoa: Công nghệ Thông tin Bộ môn: Kỹ thuật phần mềm MH: TH Lập trình hướng đối tượng	BÀI 1. LÀM QUEN MÔI TRƯỜNG VISUAL STUDIO VÀ NGÔN NGỮ C#	
--	--	--

A. MỤC TIÊU:

- Làm quen với IDE Visual Studio và ngôn ngữ lập trình C#.
- Thực hiện các thao tác cơ bản Visual Studio và viết chương trình bằng ngôn ngữ C#: tạo mới Solution, Project, nhập xuất dữ liệu, viết các biểu thức, câu lệnh cơ bản.
- Thực hiện một số bài tập cơ bản.

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SV:

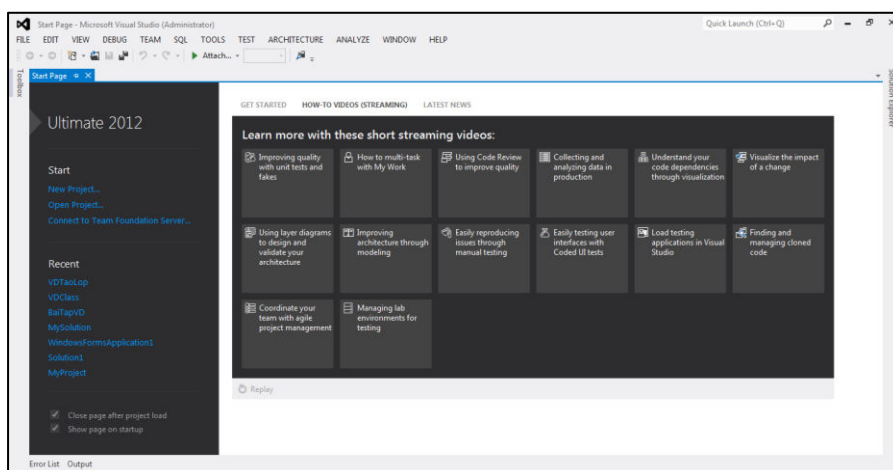
STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

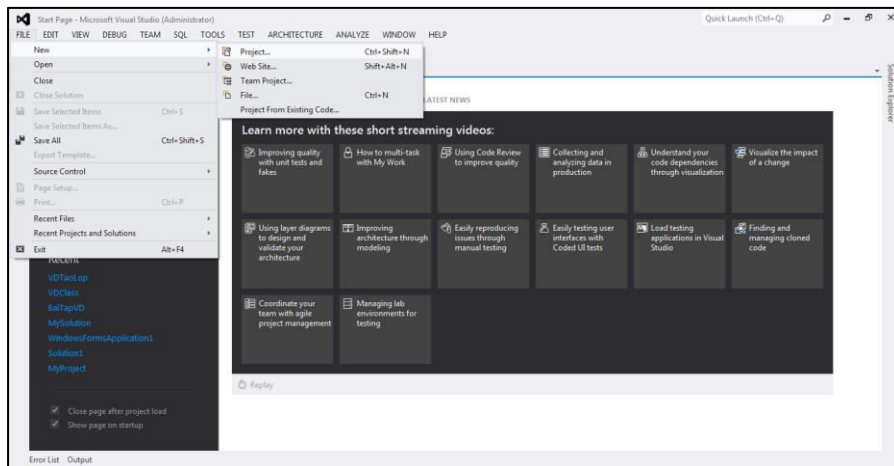
1. Tóm tắt lý thuyết

a. Tạo mới Solution và Project chạy bằng ngôn ngữ C#, biên dịch dạng Console trên Visual Studio.

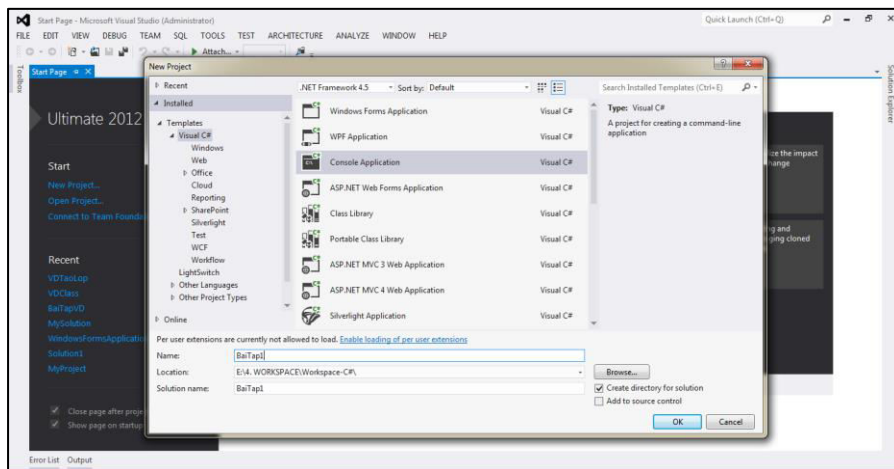
- Bước 1: Mở chương trình Visual Studio



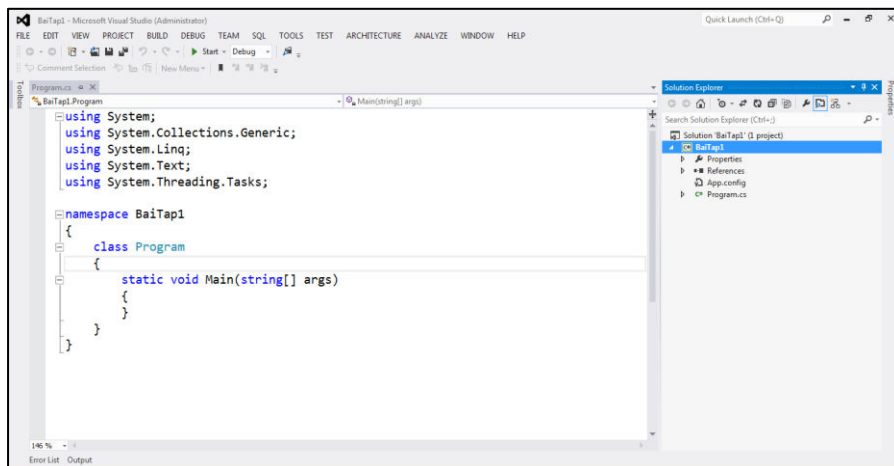
- Bước 2: Tạo mới Solution và Project



- Bước 3: Chọn loại và đặt tên cho project



- Bước 4: Viết chương trình trên project đã tạo



- b. Các lệnh khai báo thư viện và nhập xuất dữ liệu trong C#

- Khai báo thư viện:

`using <Tên thư viện>;`

- Lệnh xuất dữ liệu:

`Console.WriteLine(<dữ liệu cần xuất và các định dạng>;`

- Lệnh nhập dữ liệu dạng chuỗi:

`string Console.ReadLine()`

- Lệnh nhập dữ liệu dạng khác chuỗi: cần chuyển dữ liệu nhập vào về đúng kiểu bằng lệnh Parse.

Ví dụ: Nhập giá trị cho biến int a:

```
int a = int.Parse(Console.ReadLine());
```

c. Ví dụ các định dạng dữ liệu xuất ra của lệnh Console.WriteLine()

```
using System;
class Sample
{
    enum Color {Yellow = 1, Blue, Green};
    static DateTime thisDate = DateTime.Now;
    public static void Main()
    {
        Console.Clear();

// Format a negative integer or floating-point number in various ways.
        Console.WriteLine("Standard Numeric Format Specifiers");
        Console.WriteLine(
            "(C) Currency: . . . . . {0:C}\n" +
            "(D) Decimal:.. . . . . {0:D}\n" +
            "(E) Scientific: . . . . . {1:E}\n" +
            "(F) Fixed point:.. . . . . {1:F}\n" +
            "(G) General:.. . . . . {0:G}\n" +
            "    (default):.. . . . . {0} (default = 'G')\n" +
            "(N) Number: . . . . . {0:N}\n" +
            "(P) Percent:.. . . . . {1:P}\n" +
            "(R) Round-trip: . . . . . {1:R}\n" +
            "(X) Hexadecimal:.. . . . . {0:X}\n",
            -123, -123.45f);
// Format the current date in various ways.
        Console.WriteLine("Standard DateTime Format Specifiers");
        Console.WriteLine(
            "(d) Short date: . . . . . {0:d}\n" +
            "(D) Long date:.. . . . . {0:D}\n" +
            "(t) Short time: . . . . . {0:t}\n" +
            "(T) Long time:.. . . . . {0:T}\n" +
            "(f) Full date/short time: . . {0:f}\n" +
            "(F) Full date/long time:.. . {0:F}\n" +
            "(g) General date/short time:.. {0:g}\n" +
            "(G) General date/long time: . {0:G}\n" +
            "    (default):.. . . . . {0} (default = 'G')\n" +
            "(M) Month:.. . . . . {0:M}\n" +
            "(R) RFC1123:.. . . . . {0:R}\n" +
            "(s) Sortable: . . . . . {0:s}\n" +
            "(u) Universal sortable: . . . {0:u} (invariant)\n" +
```

```

        "(U) Universal full date/time: {0:U}\n" +
        "(Y) Year: . . . . . {0:Y}\n",
        thisDate);
// Format a Color enumeration value in various ways.
Console.WriteLine("Standard Enumeration Format Specifiers");
Console.WriteLine(
    "(G) General:.. . . . . {0:G}\n" +
    "    (default):.. . . . . {0} (default = 'G')\n" +
    "(F) Flags:.. . . . . {0:F} (flags or integer)\n" +
    "(D) Decimal number: . . . . . {0:D}\n" +
    "(X) Hexadecimal:.. . . . . {0:X}\n",
    Color.Green);
}
}
/*

```

This code example produces the following results:

Standard Numeric Format Specifiers

```

(C) Currency: . . . . . ($123.00)
(D) Decimal:.. . . . . -123
(E) Scientific: . . . . . -1.234500E+002
(F) Fixed point:.. . . . . -123.45
(G) General:.. . . . . -123
    (default):.. . . . . -123 (default = 'G')
(N) Number: . . . . . -123.00
(P) Percent:.. . . . . -12,345.00 %
(R) Round-trip: . . . . . -123.45
(X) Hexadecimal:.. . . . . FFFFFFFF85

```

Standard DateTime Format Specifiers

```

(d) Short date: . . . . . 6/26/2004
(D) Long date:.. . . . . Saturday, June 26, 2004
(t) Short time: . . . . . 8:11 PM
(T) Long time:.. . . . . 8:11:04 PM
(f) Full date/short time: . . Saturday, June 26, 2004 8:11 PM
(F) Full date/long time:.. . Saturday, June 26, 2004 8:11:04 PM
(g) General date/short time:.. 6/26/2004 8:11 PM
(G) General date/long time: . 6/26/2004 8:11:04 PM
    (default):.. . . . . 6/26/2004 8:11:04 PM (default = 'G')
(M) Month:.. . . . . June 26
(R) RFC1123:.. . . . . Sat, 26 Jun 2004 20:11:04 GMT
(s) Sortable: . . . . . 2004-06-26T20:11:04
(u) Universal sortable: . . . 2004-06-26 20:11:04Z (invariant)
(U) Universal full date/time: Sunday, June 27, 2004 3:11:04 AM
(Y) Year: . . . . . June, 2004

```

Standard Enumeration Format Specifiers

```
(G) General:.. . . . . Green
    (default):.. . . . . Green (default = 'G')
(F) Flags:.. . . . . Green (flags or integer)
(D) Decimal number:.. . . . 3(X) Hexadecimal:.. . . . . 00000003
*/
```

d. Các lệnh điều khiển cơ bản của C#

– Lệnh if/if-else

```
if(<biểu thức logic>)
{
    //khối lệnh;
}
```

```
if( <biểu thức logic> )
{
    //khối lệnh 1;
}
else
{
    //khối lệnh 2;
}
```

– Lệnh switch-case:

```
switch( <biểu_thức_lựa_chọn> ) {
    case <biểu_thức_hằng 1>:
    {
        //khối lệnh;
        break;
    }
    case <biểu_thức_hằng 2>:
    {
        //khối lệnh;
        break;
    }
    ...
    case <biểu_thức_hằng n>:
    {
        //khối lệnh;
        break;
    }
    [default : khối lệnh; break;]
}
```

– Lệnh for/while/do-while/ foreach

```
for(<biểu thức 1>; <biểu_thức_logic>; <biểu thức 2>)  
{  
    //khối_lệnh;  
}
```

```
while( <biểu_thức_logic> )  
{  
    //khối_lệnh;  
}
```

```
do {  
    //khối_lệnh;  
}while( <biểu_thức_logic> );
```

```
foreach(<kiểu dữ liệu> tên_biến in biến_mảng)  
{  
    //khối_lệnh;  
}
```

e. Mảng & danh sách liên kết trong C#

– **Mảng 1 chiều: cú pháp**

- + <kiểu dữ liệu>[] <tên mảng> = new <kiểu dữ liệu>[<kích thước mảng>];
- + <kiểu dữ liệu>[] <tên mảng> = new <kiểu dữ liệu>[<kích thước mảng>] {danh sách phần tử mảng};

Ví dụ: `int[] a = new int[4] {3, 6, 8, 1};`

– **Mảng 2 chiều: cú pháp**

- + <kiểu dữ liệu>[,] <tên mảng> = new <kiểu dữ liệu>[<số dòng,số cột>];
- + <kiểu dữ liệu>[,] <tên mảng> = new <kiểu dữ liệu>[<số dòng,số cột>] {danh sách phần tử mảng};

Ví dụ: `int[,] a = new int[2, 3] { {1,2,3},{4,5,6} };`

– **Mảng động ArrayList**

- + Là mảng động, mỗi phần tử là một object → các phần tử trong ArrayList có thể khác kiểu dữ liệu.
- + Thuộc Namespace: [System.Collections](#)
- + Khai báo:

```
ArrayList <tên mảng> = new ArrayList();  
ArrayList <tenmang> = new ArrayList (Số phần tử);
```

Ví dụ:

```
ArrayList a = new ArrayList();
```

```

        a.Add("Main");
        a.Add('a');
        a.Add(5);
        a.Add(DateTime.Now);
        for (int i = 0; i < a.Count; i++)
            Console.WriteLine("a[i]: {0}", a[i]);

```

– Danh sách liên kết – List

- + Là danh sách liên kết, các phần tử có cùng kiểu dữ liệu
- + Thuộc **Namespace**: System.Collections
- + Khai báo:

```

List<Kiểu dữ liệu> <tên mảng> = new List<kiểu dữ liệu>();
List<Kiểu dữ liệu> <tên mảng> = new List<kiểu dữ liệu>(Số phần tử);

```

Ví dụ:

```

List<int> t = new List<int>();
t.Add(5); //thêm 1 phần tử
t.AddRange(new int[] {3,9,8,2}); /* thêm nhiều phần tử là mảng 1 chiều
các số int */
foreach (int a in t)
    Console.Write(a); //a duyệt qua các phần tử của t

```

2. Bài tập thực hành trên lớp

Lưu ý: Tổ chức các bài tập chung trong một project, có hiển thị menu cho người dùng lựa chọn.

Bài tập có hướng dẫn

Bài 1. Viết chương trình nhập vào 2 số nguyên. Xuất ra tổng và hiệu, tích, thương của 2 số đó.

Hướng dẫn: file Program.cs

```

namespace BaiTap1
{
    class Program
    {
        static void Main(string[] args)
        {
            int a, b;
            Console.WriteLine(" Nhập gia tri so thu 1: ");
            /*Nhập giá trị cho biến số nguyên a nên dùng lệnh int.Parse(...) để
chuyển chuỗi nhập vào về dạng số int*/
            a = int.Parse(Console.ReadLine());
            Console.WriteLine(" Nhập gia tri so thu 2: ");
            b = int.Parse(Console.ReadLine());
            Console.WriteLine("Tong 2 so la: {0}", a + b);

```

```

        /*{0}: định dạng qui định vị trí và thứ tự của giá trị biến được xuất
ra*/
        Console.WriteLine("Hiệu 2 số là: {0}", a - b);
        //Chúng ta có thể xuất ra nhiều biến trong 1 lệnh
        Console.WriteLine("Tích 2 số là: {0} \n Thương của 2 số là: {1}", a *
b, a/b );
        /*{0}: định dạng qui định vị trí và thứ tự của giá trị a*b, {1}: tương
tự dành cho a/b */
        Console.ReadLine(); //Dừng màn hình
    }
}

```

Bài 2. Viết chương trình nhập vào thông tin của một sinh viên gồm:

- + Mã số sinh viên (string)
- + Họ tên (string)
- + Điểm trung bình - ĐTB (float)

Sau đó tính kết quả xếp loại của sinh viên, biết rằng:

- + ĐTB \geq 8.0: loại giỏi
- + $8 > \text{ĐTB} \geq 6.5$: loại khá
- + $6.5 > \text{ĐTB} \geq 5.0$: loại trung bình
- + ĐTB < 5 : loại yếu kém.

Xuất đầy đủ thông tin của sinh viên.

Hướng dẫn:

```

namespace BaiTap2
{
    class Program
    {
        static void Main(string[] args)
        {
            string mssv, hten;
            float dtb;
            string xeploai;
            Console.WriteLine("Nhập thông tin của sinh viên: ");
            Console.WriteLine("Mã số sv: ");
            mssv = Console.ReadLine(); /* mssv kiểu chuỗi nên không cần gọi lệnh
Parse */
            Console.WriteLine("Họ tên sv: ");
            hten = Console.ReadLine(); /* mssv kiểu chuỗi nên không cần gọi lệnh
Parse */
            Console.WriteLine("Điểm trung bình: ");
            /* dtb có kiểu số nên cần dùng lệnh Parse để chuyển string về đúng kiểu
*/
            dtb = float.Parse(Console.ReadLine());

```



```

        if (dtb >= 8)
            xeploai = "gioi";
        else if (dtb >= 6.5)
            xeploai = "kha";
        else if (dtb >= 5)
            xeploai = "trung binh";
        else
            xeploai = "yeu kem";
        Console.WriteLine("Thông tin sv: \n MSSV:{0} - Ho ten:{1} - ĐTB: {2:0.00} - Xep loai: {3}", mssv, hten, dtb, xeploai);
        Console.ReadLine();
    }
}
}

```

Bài 3: Tạo dãy các số chứa n số nguyên bằng cách sinh ra số ngẫu nhiên, với n do người dùng nhập vào. Sau đó xuất dãy số ra màn hình.

Hướng dẫn:

```

namespace BaiTap3
{
    class Program
    {
        static void Main(string[] args)
        {
            int n;
            Console.Write("Nhập số phần tử dãy số: ");
            n = int.Parse(Console.ReadLine());
            List<int> dso = new List<int>(n);
            Random a = new Random(); // sinh số nguyên ngẫu nhiên
            /* Sinh số ngẫu nhiên và thêm vào dãy số, dùng lệnh for để duyệt List */
            for (int i = 0; i < dso.Capacity; i++)
            {
                int k = a.Next(100); //sinh số nguyên dương ∈[0;100)
                dso.Add(k); //them số vừa sinh vào dãy số
            }
            //In dãy số, dùng foreach để duyệt List
            foreach (int t in dso)
                Console.Write(t + " ");
            Console.ReadLine();
        }
    }
}

```

Bài tập sinh viên tự làm

Bài 4. Viết chương trình biện luận và giải phương trình bậc 1.

Bài 5. Viết chương trình biện luận và giải phương trình bậc 2.

Bài 6. Viết chương trình nhập vào giá ngày/tháng/năm của một ngày trong một năm bất kỳ. Cho biết ngày đó thứ mấy. Biết rằng công thức tính thứ của một ngày/tháng/năm như sau:

Điều kiện:

Tháng < 3: tháng = tháng + 12; năm = năm – 1

Tháng >= 3

$$n = (\text{ngày} + 2 * \text{tháng} + (3 * (\text{tháng} + 1))) / 5 + \text{năm} + (\text{năm} / 4) \% 7$$

với n là kết quả thứ theo thứ tự: 0 là chủ nhật, 1 là thứ 2, ..., 6 là thứ 7.

Bài 7. Viết chương trình nhập vào 1 số nguyên n.

- Phân tích n ra thừa số nguyên tố.
- Cho biết n có bao nhiêu chữ số.

3. Bài tập về nhà


Bài 8. Viết chương trình cho nhập vào số phải là số chính phương, xuất số vừa nhập ra màn hình.

Bài 9. Nhập vào 1 ngày tháng năm. Cho biết ngày trước đó và ngày hôm sau là ngày mấy.

Bài 10. Tạo dãy các số nguyên ngẫu nhiên có 20 số (dùng ArrayList hoặc List). Mỗi phần tử là số nguyên dương và nhỏ hơn 1000. Thực hiện các yêu cầu sau trên dãy số:

- Xuất dãy số.
- Đảo dãy số
- Tìm trong dãy số có chứa số x không? Với x nhập từ bàn phím
- Xuất các phần tử có 2 chữ số.
- Xuất các số có các chữ số đều là số chẵn.
- Xuất các số nguyên tố trong dãy số (nếu có).
- Xóa khỏi dãy số các số lẻ và là bội của 3.
- Tăng giá trị lên 1 đơn vị cho các số mà nhỏ hơn 2 số liền kề (trước/và sau) nó.
- Sắp xếp dãy số tăng dần, giảm dần.

--HẾT--

Trường: ĐH CNTP TP.HCM Khoa: Công nghệ Thông tin Bộ môn: Kỹ thuật phần mềm MH: TH Lập trình hướng đối tượng	BÀI 2. LỚP VÀ ĐỐI TƯỢNG	
--	--------------------------------	---

A. MỤC TIÊU:

- Cài đặt được lớp có các thuộc tính, phương thức khởi tạo và các hàm xử lý cơ bản, phương thức static.
- Gọi được các đối tượng trong phương thức chương trình.

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SV:

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

1. Tóm tắt lý thuyết

1.1. Cú pháp tạo lớp

```
public class <Tên_lớp>
{
    //thành phần dữ liệu
    private <kiểu dữ liệu> <tên thuộc tính>;
    ...
    //thành phần phương thức
    public <kiểu trả về> <tên phương thức>([ds tham số])
    {
        //phần thân phương thức
    }
    ...
}
```

1.2. Khai báo đối tượng

```
<Tên lớp> <tên đối tượng> = new <Tên lớp>();
<Tên lớp> <tên đối tượng> = new <Tên lớp>(<ds tham số>);
```

1.3. Truy cập phương thức của lớp

```
<Tên đối tượng>.<Tên phương thức>([ds tham số]);
```

1.4. Property

```
public <Kiểu_trả_về> <Tên_Property>
{
    get
    {
        //các câu lệnh ...
        return <biểu thức trả về>;
    }
}
```

```

    set
    {
        //các câu lệnh ... xử lý value
        <tên thuộc tính> = value;
    }
}

```

1.5. Phương thức khởi tạo

a. Khởi tạo không tham số

```

public <Tên_lớp>()
{
}

```

b. Khởi tạo có tham số

```

public <Tên_lớp>(Danh sách tham số)
{
}

```

c. Khởi tạo sao chép

```

public <Tên_lớp>(<Tên_lớp> <đối tượng>)
{
}

```

1.6. Phương thức hủy

```

~<Tên_lớp>()
{
}

```

⚠ **Lưu ý: Thêm mới một lớp trong project:** Click phải Project → Add → Class → đặt tên lớp mới → OK

2. Bài tập thực hành trên lớp

Bài tập có hướng dẫn

Bài 1. Xây dựng lớp mô tả hình tròn (HìnhTron) chứa thuộc tính bán kính (số thực). Viết các phương thức:

- Property (get/set) với ràng buộc bán kính > 0
- Khởi tạo (không tham số và có tham số)
- Nhập/xuất thông tin cho hình tròn
- Tính chu vi, diện tích

Hướng dẫn:

```

public class HìnhTron
{

```

```

private double r;
public double R //Property với ràng buộc bán kính >0
{
    get { return r; }
    set
    {
        if (value < 0)
        {
            Console.Write("Du lieu bi loi");
            r = 0;
        }
        else
            r = value;
    }
}

public HinhTron() //Khởi tạo không tham số
{
    this.r = 0;
}

public HinhTron(double r) //Khởi tạo có tham số
{
    this.r = r;
}

public void nhap() //Phương thức nhập bán kính
{
    Console.Write("Nhap ban kinh hinh tron: ");
    this.r = double.Parse(Console.ReadLine());
}

public double tinhChuvi() //Phương thức tính chu vi
{
    return this.r * 2 * Math.PI;
}

public double tinhDienTich() //Phương thức tính diện tích
{
    return Math.Pow(this.r, 2) * Math.PI;
}

public void xuat() //Phương thức xuất
{
    Console.WriteLine("Hinh tron co ban kinh: {0:0.00},
                                chu vi: {1:0.00}, dien tich: {2:0.00}",r,
                                tinhChuvi(), tinhDienTich());
}
}

```

- Xây dựng hàm Main trong file program.cs

```
class Program
{
    static void Main(string[] args)
    {
        /* khai báo và khởi tạo đối tượng a thuộc lớp HìnhTron với phương thức khởi
        tạo không tham số */
        HìnhTron a = new HìnhTron();
        a.nhap(); //Gọi phương thức nhap()
        a.xuat(); //Gọi phương thức xuất()
        /* khai báo và khởi tạo đối tượng b thuộc lớp HìnhTron với phương thức
        khởi tạo có tham số */
        HìnhTron b = new HìnhTron(5.0f);
        Console.WriteLine("Chu vi hình tron b: {0:0.00}",
                                                                    b.tinhChuvi());
        Console.WriteLine("Dien tích hình tron b: {0:0.00}",
                                                                    b.tinhDienTich());
        Console.ReadLine(); //Dừng màn hình
    }
}
```

Bài 2. Xây dựng lớp NhanVien chứa các thông tin:

- Mã số, họ tên (kiểu chuỗi)
- Số ngày công (số nguyên) (giá trị > 0)
- Xếp loại (char (A,B,C)). Kết quả xếp loại thi đua dựa vào qui định:
 - + Số ngày công > 26 : loại A
 - + $26 \geq \text{Số ngày công} \geq 22$: loại B
 - + Số ngày công < 22 : loại C
- Lương ngày (200.000 đồng): Áp dụng cho tất cả các nhân viên

Hãy xây dựng thêm các phương thức sau:

- Property cho từng thuộc tính có kiểm tra ràng buộc theo yêu cầu nêu trên
- 3 hàm khởi tạo
- Hàm hủy
- Nhập/xuất thông tin nhân viên
- Hàm tính lương (số ngày công * lương ngày)
- Hàm tính thưởng: nếu xếp loại A thì thưởng 5% lương, loại B thưởng 2% lương và loại C không thưởng.

Hướng dẫn:

```
class NhanVien
{
    string ms, ht; // mã số, họ tên
    public string Ms
```

```

    {
        get { return ms; }
        set { ms = value; }
    }
    public string Ht
    {
        get { return ht; }
        set { ht = value; }
    }
    int nc; // số ngày công
    public int Nc
    {
        get { return nc; }
        set
        {
            if (value < 0)
            {
                Console.WriteLine("Du lieu sai");
                nc = 0;
            }
            else nc = value;
        }
    }
    public char Xl
    {
        /* Thuộc tính xếp loại không có hàm set, bởi vì giá trị xếp loại tính dựa
        theo ngày công không được nhập vào */
        get
        {
            if (nc >= 26)
                return 'A';
            else if (nc >= 22)
                return 'B';
            else
                return 'C';
        }
    }
    /* Thuộc tính lương ngày áp dụng chung cho tất cả nhân viên nên phải là
    thành phần static */
    public static int LuongNgay = 200000;
    //-----Hàm khởi tạo-----
    /* khi khởi tạo giá trị cho nhân viên chỉ khởi tạo cho 3 thuộc tính: ms,
    ht và nc */

```

```

public NhanVien()
{
    Ms = Ht = "";
    Nc = 0;
}
public NhanVien(string ht, string ms, int nc)
{
    this.Ht = ht;
    this.Ms = ms;
    this.Nc = nc;
}
public NhanVien(NhanVien a) //Khởi tạo sao chép
{
    this.Ht = a.Ht;
    this.Ms = a.Ms;
    this.Nc = a.Nc;
}
//-----Hàm hủy-----
~NhanVien()
{
}

//-----Các hàm xử lý khác-----
public void nhapTTNV()
{
    Console.Write("Nhap ma so: ");
    Ms = Console.ReadLine();
    Console.Write("Nhap ho ten: ");
    Ht = Console.ReadLine();
    Console.Write("Nhap so ngay cong: ");
    Nc = int.Parse(Console.ReadLine());
}
public void xuatTTNV()
{
    Console.WriteLine("{0}, {1}, {2}, {3}, {4}",
                                                                ms, ht, nc, Xl, tinhLuong());
}
public int tinhLuong()
{
    return nc*LuongNgay;
}
public float tinhThuong()
{

```



```

        if (X1 == 'A')
            return tinhLuong()*5/100;
        else
            if (X1 == 'B')
                return tinhLuong()*2/100;
            else return 0;
    }
}

```

Sinh viên xây dựng hàm Main để kiểm tra các hàm đã viết trong lớp NhanVien

Bài tập tự làm trên lớp

Bài 3. Xây dựng Lớp Nước giải khát gồm các thông tin:

- Tên hàng (string)
- Đơn vị tính (string)
- Số lượng (int) (>0)
- Đơn giá (float) (>0)
- Thuế VAT (float): Áp dụng chung cho các loại nước giải khát và giá trị có thể đổi theo thời gian, hiện tại là 10%.

Yêu cầu:

- Khai báo các property cần thiết

🔗 **Lưu ý:** đơn vị tính chỉ nhận 1 trong 4 giá trị: “kết”, “thùng”, “chai”, “lon”. Nếu đơn vị tính không thuộc 1 trong 4 tham số trên thì gán đơn vị tính là “kết”.

- Hàm khởi tạo
- Các hàm xử lý khác
 - + Xây dựng phương thức tính thành tiền dựa vào đơn vị tính (DVT) như sau:
 - DVT = “kết” hoặc “thùng” → Thành tiền = số lượng * đơn giá + số lượng * đơn giá * thuế VAT.
 - DVT = “chai” → Thành tiền = số lượng * (đơn giá/20) + số lượng * (đơn giá / 20)*thuế VAT.
 - DVT = “lon” → Thành tiền = số lượng * (đơn giá/24) + số lượng * (đơn giá / 24)*thuế VAT.
 - + Phương thức nhập/xuất thông tin mặt hàng giải khát.
- Viết hàm Main để kiểm tra các thành phần của lớp nước giải khát.

3. Bài tập về nhà

Bài 4. Xây dựng lớp hình chữ nhật (HCN) có các thành phần sau:

- Các thuộc tính mô tả chiều dài, chiều rộng
- 3 phương thức khởi tạo: không có và có tham số
- Phương thức tính chu vi
- Phương thức tính diện tích

- Phương thức tính đường chéo
- Phương thức nhập/xuất thông tin hình chữ nhật
- Phương thức thay đổi kích thước hình chữ nhật (viết theo dạng method overload):

```
void changeSize (int tx, int ty, int kiểu);
```

//kích thước (chiều dài, chiều rộng) HCN sẽ tăng lên thêm tx và ty nếu kiểu = 1, ngược lại chúng sẽ giảm tx, ty nếu kiểu =0

```
void changeSize(HCN a, int kiểu);
```

// kích thước HCN cộng thêm kích thước HCN a nếu kiểu=1, ngược lại chúng sẽ giảm nếu kiểu=0

- Viết chương trình nhập vào kích thước 1 HCN, xuất ra chu vi, diện tích và đường chéo.
- Nhập vào các kích thước để thay đổi HCN

Bài 5. Xây dựng lớp Time chứa các thành phần sau:

- Các thuộc tính giờ, phút, giây (cần kiểm tra giá trị nhập vào của giờ, phút và giây: $0 \leq \text{giờ} \leq 23$; $0 \leq \text{phút} \leq 59$; $0 \leq \text{giây} \leq 59$).
- 3 phương thức khởi tạo.
- Nhập/xuất thời gian theo dạng 24 giờ.
- Nhập/xuất thời gian theo dạng 12 giờ (cần có thêm thông tin AM và PM).
- Phương thức kiểm tra giờ có hợp lệ không? Biết rằng giờ hợp lệ là giờ theo hệ thống đồng hồ.
- Viết phương thức tăng/giảm giờ (Phương thức overload):
 - + `void tanggio (int sogiay);` → tăng giờ thông thường theo dạng giờ 24h
 - + `void tanggio (int sogiay, string kieuGio);` → kết quả ra dạng giờ 24 tiếng nếu kiểu giờ là 24, kết quả dạng 12h kèm AM/PM nếu kiểu giờ là 12.
 - + Tương tự cho 2 phương thức giảm giờ.

Bài 6. Xây dựng lớp chứa thông tin của các vận động viên chạy đua trong 1 cuộc đua marathon, gồm những thuộc tính và thành phần sau:

- Mã số, họ tên, số áo, thời gian bắt đầu (kiểu Giờ của bài 5), thời gian kết thúc (kiểu Giờ của bài 5).
- Ngoài ra còn có thành tích chuẩn (1:30:00): thời gian qui định thành tích phải \leq thành tích chuẩn này, nếu chạy chậm hơn thì coi như không đạt trong cuộc thi. Giá trị này áp dụng cho tất cả các vận động viên.
- 3 Phương thức khởi tạo.
- Phương thức nhập/xuất thông tin vận động viên.
- Phương thức tính thành tích (= thời gian kết thúc – thời gian bắt đầu), đổi ra dạng giờ:phút:giây
- Phương thức kiểm tra thời gian nhập vào bắt đầu và kết thúc có hợp lệ không? Giờ hợp lệ là giờ thuộc hệ thống giờ trên đồng hồ (loại 24h).

- Viết chương trình nhập/xuất thông tin vận động viên. Lưu ý: không nhập thành tích vì thuộc tính này lấy kết quả từ thời gian bắt đầu và kết thúc.

--HẾT--