Task1. **continue** ve **break** ifadeleri arasindaki ferq nedir?

Hər iki funksiyada döngülərdə istifadə olunur. Lakin fərqlilikləri var:

İndi Break komandamızın nümunələrini nəzərdən keçirək. İlk nümunədə, bu loopda hər hansı bir düyməni basaraq istifadəçi Break komandası ilə necə çıxış edə biləcəyini görmək üçün sonsuz bir döngü yaradacağıq.

static void Main(string[] args)

        {

            for(int i=1;i<=100000;i++)

            {

                if(Console.KeyAvailable==false)

                {

                    Console.WriteLine(i);

                }

                else

                {

                    break;

                }

                if(i==100000)

                {

                    i = 1;

                }

            }

İndi ikinci nümunəni nəzərdən keçirək. Bu nümunədə sadəcə, break funksiyasını istifadə edərək proqram çıxışları arasındakı fərqləri müşahidə edəcəyik və loopda əmrləri davam etdirəcəyik.

Aşağıdakı nümunədə, əgər döngədə sayma if kontrolu ilə bərabərdirsə, break funksiyası loopdan çıxmaq üçün verilir.

for (int sayi = 1; sayi <= 20; sayi++)

            {

                if (sayi == 7) /\* eğer sayı 7 oldusa\*/

                    break;   /\* döngüden çıx\*/

                Console.WriteLine("Sayı : {0}", sayi);

            }

            Console.ReadKey();

Aşağıdakı nümunədə, sayı 9 olduqda Continue istifadə olunur. Break komandasından fərqli olaraq 9 nömrə yazılmadı, amma loop davam etdi.

for (int sayi = 1; sayi <= 20; sayi++)

            {

                if (sayi == 9) /\* əgər sayı 9 oldusa\*/

                    continue;   /\* davam et. 9 u yazma\*/

                Console.WriteLine("Sayı : {0}", sayi);

            }

            Console.ReadKey();

Task2. **this** ifadesi static metod daxilinde istifade oluna bilermi? Aciqlayin

This metodu class in daxilində  2 dənə eyni adlı property  yarananda onları əlaqələndirmək üçündür.

This metodundan parametrli metodlarda istifadə oluna bilər.Parametrli metodun daxilində yaratdığımız propertyləər public tipli propertylərlə eyni adda olanda, parametrli metoddakı property ilə public propertyni əlaqələndirmək üçün istifadə edirik .This metodu classa aiddir obyekte yox.Məsələn:

class Human{

public string name;

public string surname;

public int num;

public void Print (string name,string surname){

[this.name](http://this.name/) = name;

this.surname = surname;

this.num = num;

}

}

Task3. **string** ve **String** keywordleri arasindaki benzer ve ferqli cehetleri yazin ve harda istifade olunabileceklerine dair kod numunesi gosterin

Hər iki keyvord ayrı ayrı proqramlaşdırma dillərində istifadə olunur. String başqa bir dilə string isə başqa bir dilə aiddir. Əgər bunu bir veb saytda parol üçün istifadə edərsək , dilin xüsusiyyətindən asılı olaraq dəyişə bilər.Yəni, bir dildə hamısı böük hərf yada kiçik hərf istəyə bilər və yaxudda başqa.

Task4. Equals(),GetHashCode(),GetType(),ToString() metodlari hansi klassin methodlaridir ve vezifeleri nedir?

Equals(),GetHashCode(),GetType(),ToString() bu metodlar constructorun metodlarıdır.Constructor class qəlibindən obyekt yarananda avtomatik olaraq işə düşür ve motor funksiyasini daşıyır. Canstructor yaranan anda obyekt yuxarıdakı 4 dənə metoda malik olur.

Equals() – gostərilən obyekti müvcud obyektə bərabər olub olmadığını göstərir.

GetHashCode() - varsaıilan hash funksiyasına xidmət edir.

ToString() -  metodu yazilarla işləməyə kömək edir.

GetType() - cari nümunənin nönünü alır

Constructor obyekt yaranan anda işə düşür ve əslində bir metoddur, amma yalnız bir funksiyani - obyekt yaratmaq funksiyasını yerinə yetirdiyi üçün void -siz işlənir.

Biz class daxilində constructor çağırmasaq belə o arxa fonda default consturctor olaraq avtomatik işə düşür ve obyekti yaradır ve başlanğıc olaraq dörd dənə funskiyası var:

GetHashCode() , ToString() , GetType() ve Equals()

Task5. **public** ve **static** keywordleri arasinda benzerlik ya da ferqlilik varmi? Varsa nelerdir?

public bize imkan verir ki yaratdigimiz poperty ve ya metgodu kod blokundan kenarda da cagiraq istifade ede bilek

staticde ise hemin kod blokunda olan propertylerin vakue larini deyishmek olmur

ve static methodu cagirmaq uchun biz yaratdigimiz obyektin sonuna noqte qoyub cagira bilerik

staticde olan butun kodlar sabit qalir

obyektin static methodunu cagiranda biz propertynin sabit qiymetini alacayiq

static methodu daxilinde olan propertyler hemin kod blokundaan kenarda istifade oluna bilmez

Task 6. OOP konseptlerinden olan **inheritance** konseptini yararli olabileceyini isbatlayacaq kod numunelerini yazin

Inheritance (miras) obyekt yönümlü proqramlaşdırmanın 3 təməl prinsipindən biridir. Digər iki prinsip Encapsulation və Polymorphismdir. Miras anlayışına daha dəqiq tərif versək , bir class metodlarını başqa bir classda rahatlıqla istifadə edə bilərik.

Buna uyğun belə bir misal verək :

Bir universitetde müəllim və tələbələrin data bilgiləri yığılmalıdır. Hər bir müəllim və tələbənin arasında bir fərd kimi özünə məxsus xüsusiyyətləri olsa da, oların bir şəxs kimi ortaq xüsusiyyyətləridə olacaqdır. Bu ortaq xüsusiyyətləri İnheritance adlandırdığımız prinsipdən istifadə edərək yararlanacayıq.

**Şəxsin xüsusiyyətləri**

Ad

Soyad

Yaşadığı Şəhər

Yaş

**Müəllim**

Ad

Soyad

Yaşadığı Şəhər

Yaş

Başa vurduğu universitet

İxtisası

**Tələbə**

Ad

Soyad

Yaşadığı Şəhər

Yaş

Fakültəsi

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;

namespace ConsoleApplication3  
{  
class shexs\_xususiyyetleri  
{  
public string Ad;  
public string Soyad;  
public string yasadigiseher;  
public int Yas;  
}

class Muellim : shexs\_xususiyyetleri   
{  
public string basaVurduguUniversitet;  
public string ixtisasi;  
}

class Telebe : shexs\_xususiyyetleri   
{  
}

class Program  
{  
static void Main(string[] args)  
{  
Muellim eyvazov = new Muellim();  
Console.WriteLine(“\t\Muellim melumatlari yuklenir\n”);

eyvazov.Ad = “Elgiz ”;  
eyvazov.Soyad = “Eyvazli”;  
eyvazov.ixtisasi = “hüquq”;  
eyvazov. basaVurduguUniversitet = “DGKA”;  
eyvaov.Yas = 19;  
eyvazov. yasadigiseher = “Baki”;

Console.WriteLine(eyvazov.Ad);  
Console.WriteLine(eyvazov.Soyad);  
Console.WriteLine(eyvazov.ixstisasi);  
Console.WriteLine(eyvazov. basaVurduguUniversitet);  
Console.WriteLine(eyvazov.Yas);  
Console.WriteLine(eyvazov. yasadigiseher +”\n\n”);

Console.WriteLine(“\t\tTelebe melumatlari yuklenir \n”);  
Telebe x = new Telebe();  
x.Ad = “Namelum”;  
x.Soyad = “Namelimov”;  
x. yasadigiseher = “Baki”;  
x.Yas = 22;

Console.WriteLine(x.Ad);  
Console.WriteLine(x.Soyad);  
Console.WriteLine(x.Yas);  
Console.WriteLine(x. yasadigiseher);

Console.ReadLine();

}}  
}

Task 7. Method overloading nedir? Harda istifade oluna biler? Istifadeye yararli oldugunu isbat edebileceyiniz kod numuneleri gosterin

Metodlar çox yüklənə bilər, yəni, çox yüklənmiş və eyni adlandırılmış üsullarla birdən çox müəyyən edilə bilər.Bular eyni adlandırıldığı üçün fərqlənməsidə çətin olacaq.Lakin, imza anlayışı deyilən bir məhfum var.Məhz bu anlayış nəticəsində biz bu metodları fərqləndirə biləcəyik. Bir metodun çox yüklənməsinə ehtiyac varsa, adı əvvəlcə eyni olmalıdır. Əks halda, bu metodun çox yükləndiyini nəzərdə tutmur, fərqli bir metodun yarandığına dəlalət verir. Eyni ad altında metodları aşdığımız müxtəlif hallar parametrlərin sayı və bu parametrlərin növləridır. Bir metodun geri qaytarılma növü bu imza konsepsiyasına daxil edilmir. Bununla bağlı sadə bir nümunə göstərək :

static int qiymetortalamasi(int sayi1,int sayi2)

        {

            int netice = (sayi1 + sayi2) / 2;

            return netice;

        }

        static int qiymetortalamasi (int sayi1, int sayi2,int sayi3)

        {

            int netice = (sayi1 + sayi2+sayi3) / 2;

            return netice;

        }

        static void Main(string[] args)

        {

            Console.WriteLine(qiymetortalamasi (13, 5));

            Console.WriteLine(qiymetortalamasi (13, 8, 7));

            Console.ReadKey();

Task 8. **variable** ve **property** arasindaki benzer ve ferqli ceheteri izah edin.

Variable və Property hər ikisidə dəyərdir. Hər ikisidə dəyər olmasına baxmayaraq arasında bir sıra fərqlər var :

1. Property classın dəyəridir.Variable methodun
2. Property təyin edildiyi classdakı bütün methodlara aid olur. Variable lakin təyin edildiyi bir methoda

Task 9. Visual Studio IDE si yoxdursa PC-nizde alternativ olaraq ne istifade edebilersiniz? Niye biz C# kodlarini Visual Studioda yaziriq? Izah edin.

Microsoft Visual Studio Microsoft-un inteqrasiya edilmiş inkişaf mühiti (IDE). Kompüter proqramları, veb-saytlar, veb proqramlar, veb xidmətləri və mobil tətbiqləri inkişaf etdirmək üçün istifadə olunur. Visual Studio, Microsoft Windows, Windows Forms, Windows Presentation Foundation, Microsoft Silverlight istifadə edir. Həm yerli kodu, həm də idarə olunan kodu yarada bilər.

Visual Studio, IntelliSense (kod tamamlama komponenti) dəstəkləyən bir kod redaktoru və kodu yeniləməyə daxildir. İnteqrasiya edilmiş səhv araşdırıcısı həm mənbə səviyyəsində düzəldici, həm də maşın səviyyəli bir debugger kimi işləyir. Digər quraşdırılmış vasitələr GUI tətbiqləri, veb-dizayn, sinif dizaynerləri və verilənlər bazası sxemi dizaynı üçün kod profiler, dizayn tərtibatçıları daxildir. Bütün zəruri proqram komponentlərini (Subversion kimi) asanlıqla əlavə etmək, redaktə etmək və silmək üçün güclü bir vasitədir.

Visual Studio 36 müxtəlif proqramlaşdırma dillərini dəstəkləyir və demək olar ki, hər hansı bir proqramlaşdırma dilini dəstəkləmək üçün kod redaktorunu və düzəltməni istifadə etməyə imkan verir. Daxili dillər C, C ++, C ++ / CLI, Visual Basic. NET, C #, F #, JavaScript, TypeScript, XML, XSLT, HTML və CSS daxildir. Python, [8] Ruby, Node.js və M kimi digər dillərə dəstək plaginlər vasitəsilə mövcuddur. Java (və J #) keçmişdə dəstəklənmişdir.

Visual Studiodan istifadə etməyimizdə başlıca məqsəd kod yazılması işini rahatlaşdırmaq və sürətli olaraq davam etdirməkdir.Əgər visual studio kimi bir tətbiq yoxdursa bu kodu NotePadla yaza bilərik.Lakin orada bizə Visual Stuioda verilən imkanlar verilmir və VS qədər sürətli modifikasiya imkanımız olmur.