به نام خدا

دانشگاه آزاد اسلامی واحد تهران مرکزی ، مجتمع دانشگاهی آیتالله هاشمی رفسنجانی، دانشکده فنی و مهندسی نام بهزاد چیذری
شماره دانشجویی 930251337 رشته مهندسی کامپیوتر – گرایش نرم‌افزار نام استاد جناب آقای دکتر نوید هاشمی
طباء نام درس اصول طراحی کامپایلر نیمسال تحصیلی نیمسال اول 96-97 اطلاعات درس و کلاس شنبه ساعت
13:30تا15:55 – کلاس 1206 تاریخ تحویل 96/06/22

موضوع تمرین شماره ۱ – زبانهای متداول برنامه نویسی، کاربردها و محدودیتها

مقدمه

زبانهای برنامه نویسی در طی زمان تغییرات زیادی داشتهاند. در ابتدا برنامه نویسی کاملا سختفزاری و با استفاده از سوئیچها انجام میشد و بعد ها به زبان ماشین یا اصطلاحاً صفرویکی تبدیل شد و نسل اول زبان های برنامه نویسی شکل گرفت. نسل دوم زبانهای برنامه نویسی یا اسمبلی دارای دستورالعملهای نزدیک به زبان انسان بود و در سطح سخت افزار اجرا میشد. پس از آن با پیدایش زبانهایی نظیر کوبول، فورترن، پرولوگ و... کار برای برنامهنویسان، بدلیل نزدیکی این زبانها به محاوره انسان، برای محاسبات آسانتر شد. پس از پیدایش شیءگرایی تحولی عظیم در برنامه نویسی شکل گرفت و زبانهای آشنای و متدوالی نظیر سیپلاسپلاس ،جاوا، سیشارپ و ... معرفی شدند.

زبانهای برنامه نویسی به دو دسته سطح بالا و سطح پایین تقسیم میشوند به این معنا که زبانهایی که در نوشتار خود دارای دستورالعمل Instruction هستند سطح پایین و زبانهایی که در نوشتار خود دارای گزاره Statement هستند سطح بالا شناخته میشوند.

در ادامه مطالب به بررسی کاربردها و محدودیت های چند زبان متداول سطح بالا میپردازیم.

سیشارپ C#

زبان برنامه نویسی سیشارپ یک زبان ساخت یافته، شیءگرا و تاثیرگرفته از زبان C++ میباشد که توسط شرکت مایکروسات در سال 2000 عرضه شد. این زبان از زبانهای تحت چارچوب .net Framework داتنت میباشد .

- سیشارپ با اهداف زیر معرفی شد:
  - سادگی، مدرن بودن و شیءگرایی ○ توسعه نرمافزارهای کاربردی
  - راحتی مهاجرت به یا از زبانهای سطح بالای دیگر مانند C++ یا جاوا
  - قدرتمندی، پایداری و کارایی بالا در عین مصرف بالای منابع)حافظه و پردازنده(

از برجستهترین ویژگیهای آن:

- زبان سیشارپ کاملا شیءگرا بوده به همین دلیل هیچ متغیر یا متد سراسری Global وجود ندارد، همه متغیرها باید عضوی از یک کلاس باشند.

```
public class Class1
{
    public Class1()
    {
    }
 }
```

- اشاره به یک خانه از حافظه [که در C++ با * نمایش داده میشد] میبایست در بلاکهای unsafe قرار گیرد
- توابع در حالت عادی فراخوانی مقداری Call by Value میباشند؛ برای فراخوانی با مرجع Call by Reference [که در C++ از & استفاده میشد] میبایست از کلمه کلیدی ref استفاده شود.

```
void Swap(ref int b)
 {
     a = a + b;
     b = a;
     a = a;
  }
```

- وراثت چندگانه در زبان وجود ندارد در عوض میتوان از اینترفیسهای مختلف Interface استفاده نمود.
- از مهمترین ویژگیهای میتوان به Garbage Collection خودکار اشاره نمود. بدین صورت که پس از اتمام کار یک شیء به صورت خودکار از حافظه پاک شده و نیازی به حذف دستی آن توسط برنامهنویس نیست.

امکانات زبان:

- کلاسهای **Partial** : کلاسهایی که توانایی پیاده سازی و اجرا را در بیش از یک فایل سورس داند. این امکان برای پیادهسازی کلاسهای بزرگ بسیار آسان کننده است.

```
partial public class Class1
{
//Something …
}
  partial public class Class1
  {
      } //Something else in other place(e.g. another .cs file)…
```

- انواع **Generic** : همانند مفهوم Template در زبان C++ میباشد با این تفاوت که نمونه Instance سازی توسط کامپایلر انجام نمیشود بلکه در زمان اجرای برنامه RunTime صورت میگیرد.

```
void Swap<T>(ref T a,ref T b)
{
    T temp = a;
a = b; b = temp;
}
```

- **Collection**ها : عبارت است از اینترفیسها و کلاسهایی که میتوانند مجموعههای مختلفی را درون خود مانند یک ارایه با امکاناتی مثلا نامتناهی بودن المانها، قابلیت داشتن لیستهای چند بعدی با نوع دادهای متفاوت؛ دارای متدهای متعدد برای راحتی کار

- **عبارات لاندا :** عبارات لاندا Lambda یک راه کوتاه برای نوشتن مقادیر توابع بی نام کلاس اول را فراهم میکنند.

از دیگر امکاناتی که زبان سیشارپ و همچنین برخی زبانهای دیگر دات نت دارند میتوان به موارد زیر اشاره کرد:

- برنامه نویسی تحت وب : API، MVC،ASP.Netها ○ ارائه وب سرویس های SOAP و REST در قابلیت - WCF ○ نوشتن برنامههای کاربردی دستکتاپ - Windows Forms و WPF
- نوشتن برنامههای برای پلتفرمهای مختلف مایکروسافتی و ویندوزی- UWP ○ کار کردن با سرویس های ویندوزی مانند آفیس ○ و...

- اشکالات و محدودیتهای زبان ○ همانطور که گفته شد سیشارت تحت چارچوب داتنت اجرا میشود و تنها روی پلتفرم ویندوز قابل اجراست.

همین امر برنامه نویسان را مجبور و محدود به کار بر روی یک پلترفرم خاص کرده.

البته؛ محدودیت پلتفرم تنها بر روی آخرین نسخهی Net Framwork. بود تا اینکه شرکت مایکروسافت چارچوب جدیدی به نام Net Core. را عرضه کرد که تحولی بزرگ برای این زبان و زبانهای دیگر داتنت بود .Dotnet Core میتواند روی هر پلتفرم یا هر سختافزاری اجرا شود مانند پلتفرمهای ویندوز و ویندوزفون ،لینوکس و ...

همچنین در پی آن شرکت مایکروسافت Universal applicationها را نیز عرضه نمود که قابلیت اجرای یک اپلیکیشن را بر روی تمام دستگاهها و پلتفرمهای مایکروسافت؛ تنها با یکبار انتشار publish را دارد.

جاوااسکریپت

زبان برنامه نویسی جاوااسکریپت یک زبان سطح بالا، شیءگرا و تفسیری میباشد. این زبان در ابتدا با نام موکا وبعدا به نام لایواسکریت و سپس جاوااسکریپت تغییر نام داد .این زبان برخلاف نامگذاریاش هیچ ارتباطی به زبان برنامه نویسی جاوا ندارد و عدهای دلیل آن را ترفند تجاری برای بالا بدست اوردن بخشی از بازار میدانند.

جاوااسکریپت خیلی زود توانست به مهمترین و متداولترین زبان برنامهنویسی وب در دنیا تبدیل
شود. با ظهور ایجکس(AJAX: Asynchronous JavaScript And XML) تحولی در این زبان رخ داد و طرفداران بیشتری را به خود جذب کرد.

امروزه فریمووركها و کتابخانههای متعددی برای پلتفرم وب با این زبان نوشته شده که کارهایی از قبیل؛ سادگی طراحی صفحات وب ،جابجایی راحتتر داده بین کارجو client و کارساز server و...

با بهبود شیوههای رایج برنامه نویسی در جاوااسکریپت کاربرد آن حتی در خارج از وب افزایش یافته است.

- حال به برخی از ویژگیهای این زبان و کاربردهایشان میپردازیم:

- **نوع دادهای پویا** : نوع دادهای به مقدار منسوب میشود نه به متغییر برای مثال متغیر x میتواند در یک جا عدد صحیح و در جایی دیگر یک رشته از کاراکتر ها باشد.

```
var x;
x = 10; // x = 10
x = 10 + "a"; // x = "10a"
```

- **شیءگرایی** : جاوااسکریپت یک زبان مبنی بر شیء میباشد. در جاوااسکریپت شیء، به یک آرایه انجمنی[1] اطلاق میشود. کلیدها و مقادیر این ارایه ویژگی شیء را فراهم میکنند.
  نمونه یک شیء در جاوا اسکریپت:

```
var me = { firstName: "behzad", lastName: "chizari", age: 21 };
```

- **توابع** : در جاوااسکریپت توابع علاوه بر نقش عادی خود میتوانند در نقش سازنده ظاهر شوند.
- Constructor یه شیء

[1] به نوع داده انتزاعی اطلاق میشود که از کلکسیونی از جفتهای)کلید key، مقدار value(تشکیل شده است، بطوری که هر کلید ممکن حداکثر یکبار در کلکسیون ظاهر میشود.

```
//function as cunstructor
function person(first, last, age, eye)
{     this.firstName = first;
this.lastName = last;     this.age =
age;     this.eyeColor = eye;
} var me = new person("John", "Doe", 50, "blue");
var you = new person("Sally", "Rally", 48,
"green");
```

همچنین توابع میتواند متدی برای یک شیء باشند.

```
var person = {
firstName: "John",
lastName: "Doe",
id: 5566,
    fullName: function () {
        return this.firstName + " " + this.lastName;
}
};
/******************/
var full_name = person.fullName();
```

- **استفاده از جاوااسکریپت در HTML - توابع DOM :**

```
  <script type="text/javascript">
document.write("Hello world");
function handle() {
            document.write("You entered: " +
                document.getElementById("myinput").value);
        }
    </script>

    <form>
        <input type="text" id="myinput" />
        <input type="button" onclick="handle()" value="Click here" />
</form>
```

برخی اشکالات این زبان:

- سافاری اپل، آپرا و اینترنت اکسپلورر مایکروسافت از نظر اجرای جاوااسکریپت و تکتک دستورات آن یکسان نیستند. ممکن است کدی در یکی از این مرورگرها به درستی اجرا شود و در دیگری به شکلی ناقص یا نامطلوب اجرا شود.

- **فریموورکها و کتابخانههای رایج و متداول جاوااسکریپتی:**

- **jQuery :** پروژه jQuery در سال 2006 با هدف استاندارد سازی شیوه نوشتن کدهای جاوااسکریپت را تغییر داد؛ بدین ترتیب نوشتن کدهای جاوااسکریپت سادهتر، زیباتر و از همه مهمتر برای اجرای صحیح روی مرورگرهای مختلف، استاندارد شد.

- **Node.js** : بستهی نرمافزاری متنبازی برای پلتفرمهای مختلف است تا بتوان اپلیکیشنهایی را برای سمت سرور نوشت. در این سالها کمپانیها و توسعهدهندگان بزرگی از Node.js استفاده کردهاند که از آن میان میتوان به Groupon ، PayPal و LinkedIn اشاره کرد .

مزیت و قدرت Node.js سرعت آن و همینطور مجموعهی رو به گسترش کاربران و توسعهدهندگانی است که ماژولهای جدید و کدهای مفید خود را در اختیار دیگران میگذارند.

موضوع تمرین شماره ۲ – کد اسمبلی

این برنامه رشته 'Hello World' چاپ میکند.

```
STACKSEG SEGMENT
  DW 64 DUP(?)
STACKSEG ENDS

DATASEG SEGMENT
  MSG DB 'Hello World'$
DATASEG ENDS

CODESEG SEGMENT
  ASSUME CS:CODESEG,DS:DATASEG

  MAIN PROC FAR
    MOVE AX,SEG DATASEG
    MOV DS,AX

    XOR AX,AX
    MOV AH,09H
    MOV DX,OFFSET MSG
    INT 21H

    MOV AH,4CH
    INT 21H
  MAIN ENDP
CODESEG ENDS
```

موضوع تمرین شماره ۳ – تولید حلقه ها از for

- While

```
While(condition){
      //……
      Statement;
}
```

```
For(;condition,statement){
      //……
}
```

- Foreach

```
foreach(var i in array){
      //……

}
```

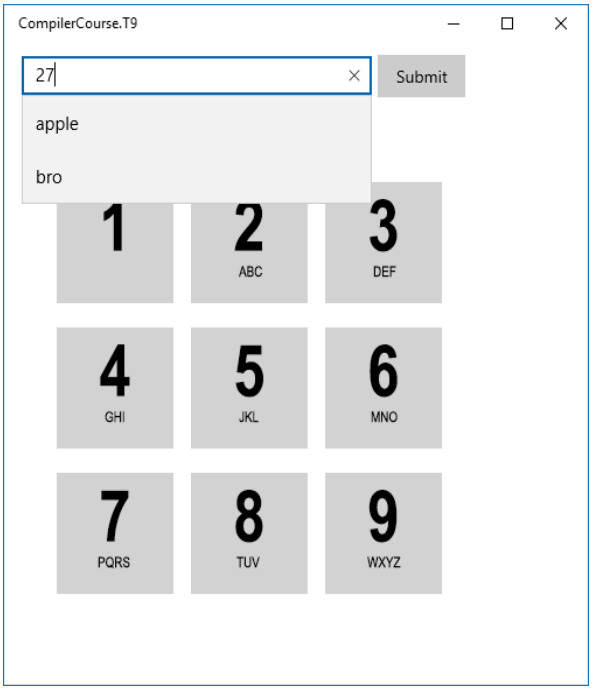
```
For(var i =0;i<array.lenght;i++){
      //……
}
```

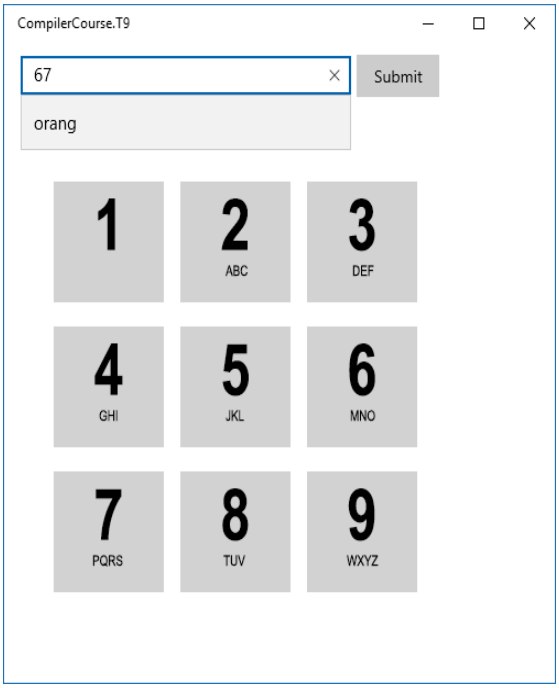
موضوع تمرین شماره۴ – برنامه T9

```
Dictionary<string, string> Words = new Dictionary<string, string>();

private string EncodeString(string word)
        {
            string result = "";
            try
            {
                // Remove digits.
                result = Regex.Replace(result, "[2-9]", string.Empty);
                // Translate to T9
                result = Regex.Replace(result, "[abc]", "2");
                result = Regex.Replace(result, "[def]", "3");
                result = Regex.Replace(result, "[ghi]", "4");
                result = Regex.Replace(result, "[jkl]", "5");
                result = Regex.Replace(result, "[mno]", "6");
                result = Regex.Replace(result, "[pqrs]", "7");
                result = Regex.Replace(result, "[tuv]", "8");
                result = Regex.Replace(result, "[wxyz]", "9");
                result = Regex.Replace(result, "[^2-9]", " ");
            }
            catch (Exception)
            {
                result = "";
            }
            return result;
        }

        //Initial Word Dictionary
        private void WordInit()
        {
            Words.Add("apple", EncodeString("apple"));
            Words.Add("orang", EncodeString("orang"));
            Words.Add("baba", EncodeString("baba"));
            Words.Add("bro", EncodeString("bro"));
            Words.Add("Hi", EncodeString("Hi"));
        }
```

پروژه به صورت فایل sln. پیوست میشود .

نتیجه برنامه :





موضوع تمرین شماره ۵ – ماشین مجازی

یک ماشین مجازی، در ابتدا توسط Popek and Goldbergبه صورت "یک نسخه کپی شده از روی یک ماشین واقعی، به صورت کارا و ایزوله شده" تعریف شد. استفاده‌های کنونی، ماشین‌های مجازی‌ای را شامل می‌شود که هیچ ارتباط با سخت‌افزار واقعی ندارند.

ماشین‌های مجازی، بر اساس استفاده و درجه ارتباط به ماشین واقعی، به دو دسته اصلی تقسیم می‌شوند. یک ماشین مجازی سیستمی یک زیرساخت محاسباتی کامل را فراهم می‌کند که از اجرای یک سیستم‌عامل کامل پشتیبانی می‌کند. در مقابل، یک ماشین مجازی فرایند، برای اجرای یک برنامه واحد طراحی شده، که این به این معناست که صرفاً از یک فرایند خاص پشتیبانی می‌کند. یک ویژگی مهم یک ماشین مجازی، این است که نرم‌افزاری که درون آن در حال اجراست، با منابع و سطوح انتزاعی که توسط ماشین مجازی اعمال می‌شود، محدود شده‌است – یعنی نمی‌تواند از دنیای مجازی خود خارج شود.

موضوع تمرین شماره ۶ – برنامه‌ی ذخیره substring ,prefix ,suffix

```
//Substrings
public static List<string> Substrings(string str)
{
    var result = new List<string>();
    for (int i = 0; i < str.Length; i++)
    {
        for (int j = 0; j < str.Length - i; j++)
        {
            var substring = str.Substring(i, j);
            if (substring == string.Empty) substring = "ε";
            result.Add(substring);
        }
    }
    return result.Distinct().ToList();
}
//Sufixes
public static List<string> Sufixes(string str)
{
    var result = new List<string>();
    var length = str.Length;
    for (int i = 0; i <= str.Length; i++)
    {
        var substring = str.Substring(i, length--);
        if (substring == string.Empty) substring = "ε";
        result.Add(substring);

    }
    return result.Distinct().ToList();
}
//Preffixes
public static List<string> Preffixes(string str)
{
    var result = new List<string>();
    for (int i = 0; i < str.Length; i++)
    {
        var substring = str.Substring(0, i);
        if (substring == string.Empty) substring = "ε";
        result.Add(substring);
    }
    return result.Distinct().ToList();
}
//Main
static void Main(string[] args)
{
    var writeFileString = $"String : banana{Environment.NewLine}";
    var substrings = Substrings("banana");
    var sufixes = Sufixes("banana");
    var preffixes = Preffixes("banana");


    writeFileString += $"-----------------------------------------------{Environment.NewLine}";
    writeFileString += $"Substrings :{Environment.NewLine}";
    foreach (var item in substrings)
    {
        writeFileString += "\t" + item + Environment.NewLine;
    }

    writeFileString += $"-----------------------------------------------{Environment.NewLine}";
    writeFileString += $"Sufixes :{Environment.NewLine}";
    foreach (var item in sufixes)
    {
        writeFileString += "\t" + item + Environment.NewLine;
    }

    writeFileString += $"-----------------------------------------------{Environment.NewLine}";
    writeFileString += $"Preffixes :{Environment.NewLine}";
    foreach (var item in preffixes)
    {
        writeFileString += "\t" + item + Environment.NewLine;
    }
    using (StreamWriter file =
         new StreamWriter($@"{AppDomain.CurrentDomain.BaseDirectory.Replace("bin\\Debug\\", "")}Substrings.txt"))
    {
        file.Write(writeFileString);
    }
}
```

نتیجه برنامه :

Substrings.txt - Notepad

File Edit Format View Help

```
String : banana
-----------------------------------------------
Substrings :
        ε
        b
        ba
        ban
        bana
        banan
        a
        an
        ana
        anan
        n
        na
        nan
-----------------------------------------------
Sufixes :
        banana
         anana
          nana
           ana
            na
             a
             ε
-----------------------------------------------
Preffixes :
        ε
        b
        ba
        ban
        bana
        banan
        banana
```

موضوع تمرین شماره۷– حق تقدم اپراتورها در زبان C

Precedence	Operator	Description	Associativity
1	++ -- () [] . ->	Suffix/postfix increment and decrement Function call Array subscripting Structure and union member access Structure and union member access through pointer (*type*){*list*} Compound literal(C99)	Left-to-right
2	++ -- + - ! ~ (*type*) * & sizeof _Alignof	Prefix increment and decrement Unary plus and minus Logical NOT and bitwise NOT Type cast Indirection (dereference) Address-of Size-of[note 1] Alignment requirement(C11)	Right-to-left
3	* / %	Multiplication, division, and remainder	Left-to-right
4	+ -	Addition and subtraction	
5	<< >>	Bitwise left shift and right shift	
6	< <= > >=	For relational operators < and ≤ respectively For relational operators > and ≥ respectively	
7	== !=	For relational = and ≠ respectively	
8	&	Bitwise AND	
9	^	Bitwise XOR (exclusive or)	
10	|	Bitwise OR (inclusive or)	
11	&&	Logical AND	
12	||	Logical OR	
13[note 2]	?:	Ternary conditional[note 3]	Right-to-Left
14	= += -= *= /= %= <<= >>= &= ^= |=	Simple assignment Assignment by sum and difference Assignment by product, quotient, and remainder Assignment by bitwise left shift and right shift Assignment by bitwise AND, XOR, and OR	
15	,	Comma	Left-to-right