مشخصات زبان برنامه‌نویسی FAPLA

زبان برنامه نویسی FAPLA زبان برنامه نویسی جدید است که دانشجویان مهندسی کامپیوتر پردیس فارابی ایجاد نموده‌اند. این زبان که مخفف FArabi Programming LAnguage ساختاری شبیه به زبان‌های کلاسیک مانند C دارد. در این نوشتار مشخصات فنی این زبان شامل ساختار و دستورات شرح داده می‌شود.

واژگان

زبان FAPLA یک زبان حساس به متن نبوده مانند سایر زبان برنامه‌نویسی دیگری دارای از مجموعه از واژگان لاتین تشکیل شده است. در ادامه به ترتیب واژگان مختلف معرفی و تعریف می‌شوند:

نوع داده: نوع کلمات کلیدی هستند که جهت تعریف نوع متغیر و ماژول‌ها استفاده می‌شوند. سه نوع در این زبان تعریف شده است: string , Real, Bool که به ترتیب به معنای رشته‌ها، اعداد و گزاره‌های منطقی است.

شناسه (Identifier): یک شناسه دنباله‌ای از حروف و اعداد است که الزاما با یک حرف باید شروع شود و طول آن حداکثر ۳۲ کاراکتر است.

کلمات کلیدی: کلمات رزرو شده در این زبان شامل واژگان استفاده شده در تعریف دستورات مانند if، then, begin, end, while,... هستند. همچنین دو دستور خاص read و write نیز جز کلمات کلیدی هستند.

جداکننده: ; بعنوان جداکننده دستورات در این زبان استفاده می‌شود.space ، enter و Tab نیز می‌تواند برای جدانمودن عبارتهای داخلی یک دستور استفاده شود.

ثابتهای منطقی: true, false ثابتهای منطقی هستند.

ثابتهای رشته‌ای: هر عبارت بین دو " یک ثابت رشته‌ای است. کاراکتر Enter در ثابت رشته‌ای مجاز نیست.

ثابتهای عددی: هر عدد صحیح یا اعشاری به عنوان یک ثابت عددی شناخته می‌شود. اعداد می‌توانند در مبنای ۱۰ یا ۱۶ بیان شوند. اعداد مبنای ۱۶ با 0x در ابتدای آنهای شناخته می‌شوند. ۱۲، ۳۴،۲، ۱۲۳۲۱، 0xA23

نمونه‌هایی از ثابتهای عددی هستند. در اعداد اعشاری دنباله‌ای از اعداد به همراه یک نقطه به عنوان نماد اعشار هستند. ۰,۱ ۲,۰۰۳ و ۳۳. نمونه‌های از ثابتهای اعشاری هستند.

عملگرها: عملگرهای این زبان در زیر نمایش داده شده است.

- چهار عملگر اصلی + - / *
- عملگر توان ^
- عملگر فاکتوریل !
- عملگرهای منطقی and or not xor
- عملگرهای مقایسه <> =< => <= = < > (<> به معنای مخالف است)
- عملگر پیمانه (باقی‌مانده %)
- عملگر دستور شرطی ؟: عبارت a?b:c به این معناست که اگر گزاره a صحیح است، b و در غیر این صورت c مقدار خروجی عبارت است.

توضیحات (Comment): توضیحات یک خطی در این زبان با نماد %% و توضیحات چند خطی بین دو نماد %%% نوشته شود. توضیحات چند خطی ممکن است شامل Enter نیز باشد. توضیحات در کامپایل حذف شده و در نظر گرفته نمی‌شوند.

دستورات

دستورات این زبان در ادامه شرح داده شده‌اند. منظور از عبارت (exp) در دستورات زیر هر ترکیب مجازی از عملوندها (شامل ثابت‌ها، متغیرها و فراخوانی ماژول) و عملگرها (عملگرهای ریاضی، منطقی و رشته‌ها) است. بعد از پایان هر دستور به جز تعریف ماژول دستور شرطی و حلقه و بلاک کد لازم است ; وجود داشته باشد. بلاک کد Code Block مجموعه‌ای از دستورات پشت سر هم هستند که با begin و end محصور شده‌اند و به منزله یک دستور در نظر گرفته می‌شوند (معادل با {} در زبان C).

- **دستورات انتساب:** دستورات انتساب برای انتساب یک عبارت به یک متغیر استفاده می‌شود و به صورت id=exp است.
- **تعریف متغیر:** تعریف متغیر در این زبان به صورت id:type تعریف می‌شود که در آن id شناسه و type نوع متغیر است. متغیر یا در بدنه یک ماژول و یا در بلاکهای کد می‌توانند تعریف شود.

- **دستورات خواندن و نوشتن:** زبان از دو دستور read و write برای خواندن و نوشتن از ورودی استفاده می‌کند. دستور read x یک مقدار از ورودی خوانده و آن را در متغیر x قرار می‌دهد و دستور write exp مقدار عبارت exp را در چاپ می‌نویسد. کلمات read و write کلمات کلیدی هستند.
- **دستور شرطی:** دستور شرطی if همانند سایر زبانها برای اجرای یک دستور در صورت درستی یک شرط استفاده می‌شود. ساختار دستور به صورت

```
If exp then
        Code1
Else
        Code2
```

در صورت درستی عبارت exp، کد یک و در غیر آن کد دو اجرا می‌شوند که ممکن است یک دستورالعمل یا یک بلاک کد باشند. If، then و else کلمات کلیدی هستند. دستور شرطی ممکن است با یا بدون else باشد. به عنوان نمونه دستور زیر فاقد else است.

```
If x>y then
        Flag = true;
```

- **حلقه**: حلقه در زبان با دستور while مشخص می‌شود. ساختار حلقه به صورت زیر است:

```
While exp
        code
```

بدنه حلقه (code که ممکن است یک دستور یا یک بلاک کد باشد) تا هنگامی که عبارت exp درست باشد، اجرا می‌شود. while کلمه کلیدی است.

- **تعریف ماژول:** ماژول‌ها در این زبان مانند توابع در زبان C هستند. یک ماژول به صورت زیر تعریف می‌شود:

```
Module name
Input:
        Input Variable declarations;
Output: outputType;
Begin
        Module Codes
End
```

کلمات آبی رنگ در کد بالا کلمات کلیدی هستند. در خط اول بعد از کلمه کلیدی module نام آن ذکر می‌شوند. دو بخش بعدی از ماژول تعریف متغیرهای ورودی و خروجی ماژول است. ورودی ماژول با کلمه

کلیدی input و سپس مجموعه متغیرهای تعریف‌شده است. خروجی ماژول با کلمه کلیدی output و سپس نوع خروجی ماژول تعریف می‌شود. بدنه ماژول یک بلاک کد است. خروجی هر ماژول با کلمه کلیدی return تعریف می‌شود. کد زیر یک نمونه ماژول را نمایش می‌دهد.

```
Module Square
Input:
        X:real;
        Y:real;
Output: real;
Begin
        Return x*y;
End
```

یک ماژول ممکن است هیچ ورودی یا خروجی نداشته باشد، که در این صورت از بخش input و output صرف نظر می‌شود. دستور return در ماژول‌های بدون خروجی مجاز نیست. کد زیر یک نمونه از ماژول بدون ورودی و خروجی است:

```
Module Hello
Begin
        Write “Hello world!”;
End
```

- **فراخوانی ماژول**: فراخوانی ماژول به صورت id(params) است که در آن id نام ماژول و params ورودی‌های ماژول هستند که با ویرگول باید از هم جدا شوند. در صورتی که ماژول بدون ورودی باشد، بین پرانتزها نیاز به نوشتن عبارتی نیست. ماژول به شرطی که خروجی داشته باشد، می‌تواند در عبارتها استفاده و ترکیب شود.

تحلیل معنایی

جهت پیاده‌سازی تحلیل‌گر معنایی در این زبان شما فقط کافی دو بخش جدول نمادها و بررسی نوع را پیاده‌سازی کنید. جدول نمادها می‌بایست شناسه‌های تعریف نشده و تکراری را شناسایی کند. کامپایلر پیاده‌سازی شده از نوع تک گذره (single pass) است، به همین جهت هر متغیر و ماژول پیش از استفاده حتما باید تعریف شود.

قوانین معنایی دستورات زبان مشابه با سایر زبانهای کلاسیک است (شرط if باید از نوع bool باشد و ...). در اینجا تنها قوانین مهم یا متفاوت ذکر می‌شود. در این زبان bool<real<string، بنابراین bool قابل تبدیل به عدد و عدد قابل تبدیل به رشته است. در تبدیل ضمنی گزاره منطقی به عدد، درست معادل یک و غلط معادل صفر در نظر گرفته می‌شود. (توجه کنید که برعکس آن مجاز نیست، برخلاف زبان C).

عملگرهای مجاز برای نوع bool، فقط عملگرهای منطقی، تساوی =،مخالف <> به همراه دستور شرطی ؟ است. در مورد رشته‌ها عملگرهای مقایسه، جمع + مجاز هستند و در مورد اعداد کلیه عملگرها به جز عملگرهای منطقی قابل استفاده هستند.

دستور نوشتن می‌تواند هر نوع متغیری را از ورودی را بخواند، بنابراین جنس ورودی بسته به نوع متغیر خواهد بود. اما دستور write فقط قادر به نوشتن رشته در خروجی است. بنابراین عبارت exp در این دستور باید از نوع رشته باشد.

هر برنامه حتما باید یک ماژول به نام main داشته باشد، که ماژول اصلی برنامه و نقطه شروع اجرای آن است. main بدون ورودی و خروجی باید باشد.

یک نمونه از برنامه به زبان FAPLA (دقت کنید زبان به حروف کوچک و بزرگ حساس نیست)

```
%%%This is a sample Written in FAPLA
The program compute fibonacci serie%%%
    Module F
    Input:
    X:real;
    Output: real;
    Begin
        If x > 0 then
            Return f(x-1)+f(x-2);
        Return 1;
    End
    Module main
    Begin
        I:real;
        I=20;
        Write f(i);
```

```
end
```

مراحل انجام و تحویل پروژه

نمره‌دهی پروژه از ۱۰۰ است و در نمره نهایی به ۶ الی ۸ نمره از ۲۰ تبدیل خواهد شد. تحول پروژه در طی سه مرحله مستقل است، به این معنا که کیفیت تحویل در یک مرحله تاثیری در مرحله بعدی ندارد و نمره هر مرحله بعد از گذشت موعد آن دیگر قابل جبران نیست. جهت تحویل پروژه تعدادی فایل ورودی به شما داده خواهد شد و برنامه شما باید به ازای هر فایل خروجی صحیح را تولید کند. هر خروجی صحیح بخشی از نمره آن مرحله را تشکیل می‌دهد. به عنوان نمونه در تحویل مرحله اول ۵ فایل وجود خواهد داشت که هر یک ۳ نمره از ۱۵ نمره مرحله اول را تشکیل می‌دهد. خروجی یک فایل حتی اگر فقط بخشی از آن اشتباه باشد کلاً اشتباه در نظر گرفته می‌شود. بنابراین سعی کنید، پروژه را قبل از تحویل با در نظر گرفتن همه حالات مختلف کامل تست کنید چرا که فرصت مجددی نخواهید داشت.

مرحله اول (۱۵ امتیاز): بخش تحلیل‌گر لغوی این زبان را پیاده‌سازی کند به نحوی یک فایل با پسوند FAPLA حاوی کد برنامه دریافت کند و در خروجی دنباله توکن‌های یافت شده در آنرا چاپ کند. اگر فایل دارای خطای لغوی بود، پیام مناسب چاپ کرده اما بتواند باقی آنرا ترجمه کند (فرآیند تحلیل نباید با رسیدن به خطای لغوی متوقف شود)

مرحله دوم (۳۵ امتیاز): در این مرحله شما باید با افزودن تحلیل دستوری مرحله قبل را کامل کنید. برنامه مجددا باید فایل کد را دریافت کند و به عنوان در خروجی تنها کافی است لیستی از خطاهای دستوری را چاپ نماید. برنامه نباید با رسیدن به اولین خطا متوقف گردد و باید بتواند کل برنامه پویش کند.

مرحله سوم (۵۰ امتیاز): آخرین مرحله از تحویل پروژه پیاده‌سازی کامپایلر زبان است که شامل تحلیل‌گر معنایی و اجرای کد است. برنامه نوشته شده باید فایل کد را به عنوان ورودی دریافت کند و اگر برنامه شامل هر نوع خطای معنایی، دستوری یا لغوی است در خروجی نمایش دهد و در صورت صحت ورودی کد را اجرا کند.

راهنمایی: جهت اجرای کد لازم نیست برنامه به زبان ماشین ترجمه شود. شما می‌توانید به صورت برخط درخت AST را اجرا نمایید و یا اینکه کدی به یک زبان واسط (مانند C#) تولید و سپس آنرا اجرا نماید. به عنوان نمونه برنامه بخش قبل می‌تواند به کد زیر ترجمه شود:

```
class Program
{
```

```
        static double f(double x) {
                  if(x > 0) return f(x - 1) + f(x - 2) ;
                  return 1; }

        static void Main(string[] args)
        {
            double i;

            i = 20;
            Console.WriteLine(f(i));

        }
    }
```

امتیازات ویژه

با انجام بخش‌های زیر شما می‌توانید نمرات ویژه علاوه بر نمره اصلی پروژه کسب کنید. (نمره اصلی پروژه از ۱۰۰ بوده و با احتساب این موارد تا سقف ۱۲۵ نمره قابل افزایش است). این نمرات علاوه بر بیست در نمره نهایی شما لحاظ می‌شود. میزان تاثیر دقیق این بخش در نمره اصلی بسته به نمره نهایی در نظر گرفته شده برای پروژه بین ۱,۵ الی ۲ برای پروژه خواهد بود. تحویل این مراحل باید در آخرین مرحله از پروژه است.

ایجاد ویرایشگر زبان (۲۵ نمره): ویرایشگر زبان محیطی است که شما می‌توانید در آن همزمان با تایپ خطاهای کامپایل را مشاهده کنید (همانند محیط Visual Studio). این ویرایشگر همزمان با تایپ (توجه کنید همزمان نه با زدن یک کلید) لازم است ۱) کلمات کلیدی را تشخیص و رنگی کند و ۲) خطاهای کامپایل را به همراه خط آن نمایش دهد (یا زیر آن خط بکشد). جهت ترجمه باید از روشهای مدیریت خطا استفاده شود و عملیات ترجمه با رسیدن به اولین خطا متوقف نشود.