

معرفی زبان تسلنگ

مستند حاضر زبان ساده‌ی تسلنگ^۱ را معرفی می‌کند. در گام‌های تمرین عملی درس طراحی کامپایلر، بخش‌هایی از یک مترجم برای این زبان نوشته می‌شوند. قواعد این زبان در ادامه‌ی این مستند بیان می‌شوند.

- ۱ زبان تسلنگ دارای دو نوع داده‌ی اصلی می‌باشد:
(الف) متغیرهای نوع word یک مقدار عددی علامت‌دار را در خود نگه می‌دارند.
(ب) اشاره‌گرهای word به یک آرایه از نوع word اشاره می‌کنند.
- ۲ برنامه‌های این زبان در یک فایل نوشته می‌شوند که شامل تعدادی تابع می‌باشد. در این زبان متغیرهای جهانی (Global) وجود ندارند.
- ۳ خط اول هر تابع، نام تابع و پارامترهای آن را مشخص می‌کند.
- ۴ بدنه‌ی هر تابع بین دو علامت «{» و «}» قرار می‌گیرد و شامل تعدادی عبارت (Statement) می‌باشد.
- ۵ شباهت زیادی بین ساختار عبارت‌ها و اولویت عملگرها در زبان تسلنگ و زبان C وجود دارد.
- ۶ هر بلوک (Block) در این زبان نیز بین دو علامت «{» و «}» قرار می‌گیرد.

1 The Simple Language (of NIT Compiler Course II)

۷ در هر بلوک می توان متغیر تعریف نمود و بلوک ها می توانند تو در تو (Nested) باشند. حوزه (Scope) هر متغیر مشابه زبان C تعریف می گردد.

۸ متغیرهایی محلی هر بلوک با استفاده از کلمه ی کلیدی «local» و به شکل زیر تعریف می شوند:

```
local w;           # w is a variable of type word or word pointer
```

۹ مقدار خروجی یک تابع با استفاده از کلمه ی کلیدی «return» مشخص می شود و با اجرای عبارتی که با این کلمه شروع می شود، اجرای تابع خاتمه می یابد.

۱۰ مثالی از تعریف یک تابع در ادامه نشان داده می شود.

```
def sum(a, b)
{
    local sum;
    sum = a + b;
    return sum;
}
```

۱۱ همان طور که در مثال بعدی دیده می شود، پارامترهایی که به یک تابع داده می شوند می توانند به یک آرایه اشاره نمایند و آن تابع می تواند عناصر آن آرایه را تغییر دهد.

```
def sum_all(A, n)
{
    local sum;
    local i;
    i = 0;
    sum = 0;
    while (i < n) {
        sum = sum + A[i];
        i = i + 1;
    }
    return sum;
}
```

۱۲ هر برنامه‌ی تسلنگ باید شامل یک تابع با نام main باشد که اجرای برنامه با فراخوانی این تابع آغاز می‌گردد.

۱۳ تابع main بدون پارامتر است و یک word بر می‌گرداند که کد برگشتی برنامه را مشخص می‌نماید.

```
def main()
{
    local A;
    A = array(100);
    sum_all(A, 100);
    return 0;
}
```

۱۴ در زبان تسلنگ از عبارت شرطی if و حلقه‌ی while با ساختاری مشابه زبان C می‌توان استفاده کرد.

۱۵ مثال زیر استفاده از if را نمایش می‌دهد.

```
# Inefficient calculation of Fibonacci sequence
def fib(n)
{
    if (n < 2)
        return 1;
    return fib(n - 1) + fib(n - 2);
}
```

۱۶ در تسلنگ دو تابع کتابخانه‌ای برای خواندن ورودی و چاپ خروجی وجود دارند:
readword() مقدار یک عدد را از ورودی استاندارد می‌خواند:
printword() مقدار یک عدد را در خروجی استاندارد چاپ می‌نماید:

۱۷ در تسلنگ دو تابع کتابخانه‌ای برای تخصیص و آزاد کردن حافظه‌ی آرایه‌ها وجود دارند:
array() یک اشاره‌گر به آرایه‌ای با اندازه‌ی داده شده بر می‌گرداند:
free() آرایه‌ای که اشاره‌گر داده شده به آن اشاره می‌کند را آزاد می‌کند:

قواعد تجزیه‌ی زبان تسلنگ

در ادامه ساختار BNF زبان تسلنگ نمایش داده شده است. در این ساختار اولویت‌های عملگرها (که مشابه عملگرهای زبان C هستند) در نظر گرفته نشده است. همچنین در برنامه‌های زبان تسلنگ، علامت # و حروفی که بعد از آن آمده‌اند تا آخر خط Comment محسوب می‌شوند.

```
prog ::=  func |
         func prog
func ::=  def iden ( flist ) { body }
body ::= stmt |
         stmt body
stmt ::=  expr ; |
         def ; |
         if ( expr ) stmt |
         if ( expr ) stmt else stmt |
         while ( expr ) stmt |
         return expr ; |
         { body }
def ::=  local iden
expr ::= iden = expr |
         iden ( clist ) |
         iden [ num ] |
         expr + expr |
         expr - expr |
         expr * expr |
         expr / expr |
         expr % expr |
         expr < expr |
         expr > expr |
         expr == expr |
         expr <= expr |
         expr >= expr |
         expr || expr |
         expr && expr |
         ! expr |
         - expr |
         + expr |
         ( expr ) |
```

```

            iden |
            num
flist ::=   |
            iden |
            iden , flist
clist ::=  |
            expr |
            expr , clist
num ::=    [0-9]+
iden ::=   [a-zA-Z_] [a-zA-Z_0-9]*

```