به نام خداوند بخشنده مهربان

فاز ۳ پروژه - درس اصول و طراحی کامپایلرها دکتر رزازی ترم پاییز ۱۴۰۰-۱۳۹۹ - دانشکده کامپیوتر، دانشگاه صنعتی امیرکبیر	

لطفا قبل از شروع به حل کردن تمرین به نکات زیر توجه فرمایید:

۱) هدف از انجام تمرین‌ها، یادگیری عمیق‌تر مطالب درسی است. در نتیجه هرگونه کپی‌برداری موجب کسر نمره خواهد شد.

۲) . تمام فایل‌های خواسته شده را در یک فایل فشرده قرار دهید. نام فایل نهایی را شماره دانشجویی های خود قرار دهید. (برای مثال phase3_9531999_9631747.zip)

۳) در کنار این فایل چند ویدئو برای راهنمایی شما وجود دارد که توصیه می کنیم آن ها را مشاهده بکنید. محتوای ارائه شده در فیلم ها در آدرس زیر و courses قابل دسترسی است:
https://drive.google.com/drive/folders/121W_A0glhLA7_nPf9FjQJiexcVzq8t2v?usp=sharing

۴) در صورت وجود هرگونه سوال می توانید از طریق ایمیل با تدریسیار در ارتباط باشید.

moh.robati@aut.ac.ir

sheykh@aut.ac.ir

rouzbehghasemi1998@gmail.com

sepehr.asgarian@gmail.com

amirali.sajjadi98@gmail.com

parsafarinnia@gmail.com

شما در مرحله قبل یک تجزیه گر[1] طراحی کردید که از نشانه های برگردانده شده استفاده کرده و تشخیص داد آیا برنامه با قوانین گرامر داده شده مطابق هست یا خیر.همچنین شما گرامر داده شده را رفع ابهام کردید. حال از شما می خواهیم که کد های گرامر را پیاده سازی کنید.

نکته: گرامر داده شده را می توانید با استفاده از اولویت[2] ها و تغییر ناپایانه[3] ها ، بدون اینکه زبان پذیرنده زبان عوض شود، جهت راحت تر شدن کار و ممکن شدن پیاده سازی (مانند استفاده از وصله زنی[4]) تغییر دهید .

شکل برنامه خروجی:

برنامه c شما باید به شکل زیر باشد(قالب کد در یک فایل پیوست است که در آن جزئیات کد پرش آمده است.):

$\#include\ <stdio.h>$

$\#include\ <setjmp.h>$

$int\ array[(int)1e6]\ ;$

$jmp\ codes\ in\ template$

$int\ Id1, Id2, \ldots, Idn\ ;$

$int\ main()$

{

$statement\_1;$

$L1: statement\_2;$

.

.

$L2: statment\_k\ ;$

.

.

$statement\_n;$

}

[1] Parser
[2] Precedence
[3] Nonterminal
[4] Backpatching

دستورات قابل استفاده در main :

1) $x = y\ op\ z$	$op\ \in \{+, -, *, /, \%, \}$	$x, y, z \in Z\ \cup ID$
2) $x = op\ y$	$op\ \in \{-, !\}$	$x, y\ \in Z\ \cup ID$
3) $x = y$		$x, y \in Z\ \cup ID$
4) $goto\ L$		$L\ \in \{L1, \dots, Ln\}$
5) $if\ (x\ relop\ y)\ goto\ L$	$relop\ \in \{<, >, <=, >=, ==, ! =\}$	$x, y \in Z\ \cup ID$
6) $array[x] = y$		$x, y \in Z\ \cup ID$
7) $x = array[y]$		$x, y \in Z\ \cup ID$
8) $printf("\%d", x)$		$x \in\ ID$
9) $forward\_jmp(x)$		$x \in Z\ \cup ID$
10) $back\_jmp(x)$		$x \in Z\ \cup ID$
11) $return\ 0$		

نکته اول:قالب خاصی برای اسم ID ها نیاز نیست.صرفا مهم است که با قواعد زبان c سازگار باشد.

نکته دوم: شما در اینجا باید استفاده اعداد صحیح در جایگاه عبارات بولی و همچنین عبارات بولی در محاسبات اعداد را مانند زبان c پیاده سازی کنید.همچنین می توانید از این ویژگی زبان c در کد پیاده سازی شده استفاده کنید. به طور مثال:

$$x \in Z, y = true\ \rightarrow\ T = y + y + x = 2 + x$$

نکته سوم:استفاده از 2 برچسب[5] پشت هم برای اشاره به یک خط مجاز نیست.هر خط حداکثر یک برچسب می تواند در پشت خود داشته باشد.

نکته چهارم:اجازه تعریف آرایه دیگری غیر از array اصلی که در شکل برنامه خروجی توضیح داده شده است را ندارید.

نکته پنجم: لطفا از دستورات و عملگر ها خارج از قالب چیز هایی که در صفحه قبل توضیح داده شده اند استفاده نکنید زیرا نمره ای تعلق نخواهد گرفت.

نکته ششم:در صورت کامپایل شدن و دادن خروجی درست برنامه شما نمره دارد.

نکته هفتم: یک کد نمونه به پیوست آمده است که روش استفاده و معنی قواعد در آن آمده است.(توجه داشته باشید که کامنت ها صرفا جهت رفع ابهام است و جزئی از زبان نیست.)

نکته هشتم: یک کد دیگر به پیوست آمده است که روش استفاده از 2 دستور پرش را نشان داده است.

[5] Label

نکته نهم: متغییر float در کد c نخواهیم داشت و مقادیر توکن های floatnumber را به int در خود پایتون cast کنید.

نکته دهم: برای آسان سازی پروژه نیازی به بررسی نوع ها[6] و تحلیل معنایی[7] نیست.(یعنی فرض کنید هر کدی که از مرحله تحلیل گر نحوی با موفقیت گذشت است درست می باشد و صرفا به تولید کد فکر کنید.)

به شما تستی نخواهد داده شد که نیاز به بررسی این موضوعات(مانند تقسیم بر ۰ یا پارامتر اشتباه یا استفاده از تابع void در exp) داشته باشد.

نکته یازدهم: خروجی دادن جدول نماد ها الزامی است و نیم نمره دارد. برای راهنمایی به تمرین دوم و توضیحات آقای رباطی درباره آن مراجعه کنید.

نکته دوازدهم: تولید کد به سه بخش تقسیم می شود و هر بخش 1 نمره دارد و هر بخش به صورت صفر یا یک نمره داده می شود.

بخش اول ، تولید کد عبارت های محاسباتی[8]:

در اینجا شما باید کد هایی که از عملگر های اصلی(-،*،/،+،٪) و دستور انتساب استفاده می کنند را پیاده سازی کنید.

بخش دوم ، تولید کد عبارت های بولی و شرطی:

در اینجا شما باید کد هایی که از عبارات بولی(شامل true , false , relop , not , and , or) و انتساب آن ها و حلقه ها(شامل for و while) و دستورات شرطی(شامل if,else,elseif و on,case) استفاده می کنند را پیاده سازی کنید.

بخش سوم ، تولید کد توابع:

در اینجا شما باید استفاده از توابع را پشتیبانی کنید.

[6] Type checking
[7] Semantic analysis
[8] Arithmetic expression