HW3 ۸۱۰۱۰۱۳۹۴ محمدسینا پرویزی مطلق

سوال ① از ۴ attribute استفاده می‌کنیم.

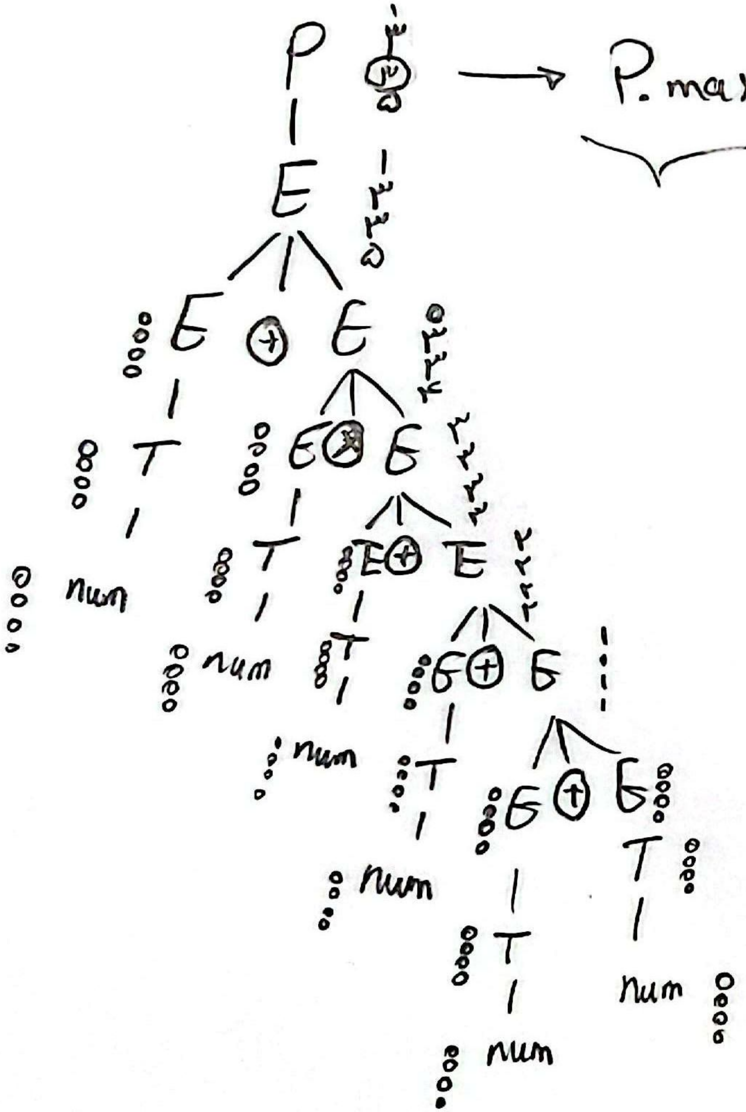
.prefix plus ← تعداد + ها سمت چپ (از چپ یا prefix عبارت + ها)

.Suffix plus ← " " " راست (از راست یا Suffix " " ")

.length ← طول رشته را مشخص می‌کند (معادل تعداد + ها)

.max ← max + ها

```
P → E { P.max = E.max }

E → E1 + E2 { if (E1.prefixplus == E1.length)
                  E.prefix plus = E1.prefixplus + E2.prefixplus + 1;
              else
                  E.prefix plus = E1.prefixplus;
              if (E2.Suffixplus == E2.length)
                  E.Suffix plus = E2.Suffix plus + E1.Suffixplus + 1;
              else
                  E.Suffix plus = E2.Suffix plus;
              E.max = max(E1.Suffix plus + E2.prefixplus + 1, E1.max, E2.max);
              E.length = E1.length + E2.length + 1; }

E1 x E2 { E.prefixplus = E1.prefixplus;
          E.Suffixplus = E2.Suffixplus;
          E.max = max(E1.max, E2.max);
          E.length = E1.length + E2.length + 1; }

T { E.prefix plus = T.prefix plus; E.Suffix plus = T.Suffixplus; E.max = T.max;
    E.length = T.length; }

T → (E) { T.prefix plus = E.prefixplus; T.Suffixplus = E.Suffixplus; E.max = T.max;
          E.length = T.length; }

num { T.prefixplus = 0, T.Suffixplus = 0, T.max = 0, T.length = 0; }
```



P.max = ۳

prefixplus
suffixplus
max
length

سوال ۲)

ب)

برای Static Scoping

مقادیر اولیه متغیرها
x = ۱۰
y = ۲۰
z = ۳۰

Inside main, x = ۱۰ , y = ۲۰ , z = ۳۰

Inside func1, x = ۵ , y = ۲۵ , z = ۳۰

Inside func2, x = ۱۰ , y = ۲۵ , z = ۱۵

Inside fun3, x = ۳۰ , y = ۲۵ , z = ۱۵

الف)

برای dynamic Scoping

مقادیر اولیه
x = ۱۰
y = ۲۰
z = ۳۰

Inside main, x = ۱۰ , y = ۲۰ , z = ۳۰

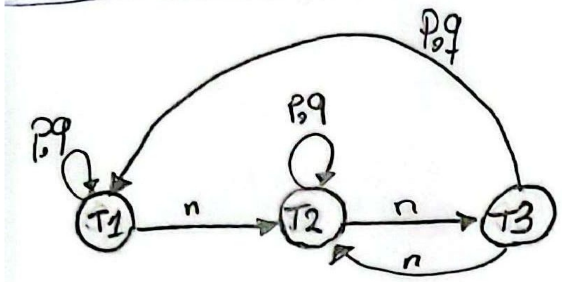
Inside func1, x = ۵ , y = ۲۵ , z = ۳۰

Inside func2, x = ۵ , y = ۲۵ , z = ۱۰

Inside func3, x = ۳۰ , y = ۲۵ , z = ۱۰

شکل چپ

سوال ۳



$P \to S$

$S \to nS \mid pS \mid n \mid p \mid q$

$n(p+q)^*n$

P → S { P.Count = S.Count; }

S → n S1 { if (S1.state == T1 || S1.state == T3)
    S.state = T2;
  else
    S.state = T3; S.Count++ ; } |

n { S.state = T2 ; S.count = 0; } |

p S1 { if (S.state == T1 || S.state == T3)
    S.state = T1;
  else
    S.state = T2; } | q S1 { if (S.state == T1 || S.state == T3)
    S.state = T1;
  else
    S.state = T2; } |

p { S.state = T1; } | q { S.state = T1; }

f() را که

سوال ۴ الف) خیر تقسیم بر صفر رخ نمی‌دهد. در تابع main تابع g() صدا زده می‌شود و در تابع g ورودی تابع g(۲) را

۲۱ دارد بر اساس آن تصمیم‌گیری می‌کنیم (~~۲~~ صدا زده می‌شود) اگر y زوج بود که دارد if می‌شویم و یکی بالا می‌شود و در نهایت x که برابر با (۲-)۱ است return می‌شود. ولی اگر فرد بود دارد else می‌شویم که $res = \frac{1}{(1+x+y)}$ محاسبه می‌شود

$1+x=1-۳=-۲$ است بنابراین اینکه مخرج صفر باشد y باید برابر با ۲+ باشد که اصلا اینجوری نمی‌تواند باشد چون اگر زوج بود دارد if می‌شود الان چون فرد است هیچ وقت تقسیم بر صفر اتفاق نمی‌افتد. در این حالت هم return ۲-x می‌شود.

ب) در حالت Dynamic Scoping اما $x=-۸$ است و حاصل $1+x=-۷$ محاسبه می‌شود و جواب اگر y برابر با ۷+ باشد تقسیم بر صفر رخ خواهد داد.

سؤال a

class X {   ← {(J, int), (x, int[3]), (Z, String), (f, int × float[10] → void)} = A

[illegible]

int j;   ← A

bool x[10];   ← A

void f(int a, float b[10]) {   ← A + {(a, int), (b, float[10])} = B

[illegible]

for (int i=0; i < a; i++) {   ← B + {(i, int)} = C

x[i] = b[i] * a < Z [b[i] + j];   ← C

}   ← B

}   ← A

String Z;   ← A

};   ← {}

ب)

$$T \vdash x[e] : \quad \frac{\dfrac{e \in dom(T)}{T \vdash e} \qquad \dfrac{x \in dom(T)}{T \vdash x}}{T \vdash x[e]}$$

ج)

$$x[i] = b[i] \times a < z[b[i] + j]$$

$$\frac{\dfrac{\dfrac{x \in dom(T)}{T \vdash x} \qquad \dfrac{\in dom(T)}{T \vdash i}}{T \vdash x[i]} \qquad \dfrac{\dfrac{\dfrac{\dfrac{b \in dom(T)}{T \vdash b} \quad \dfrac{j \in dom(T)}{T \vdash i}}{T \vdash b[i]} \quad \dfrac{a \in dom(T)}{T \vdash a}}{T \vdash b[i] \times a} \qquad \dfrac{\dfrac{z \in dom(T)}{T \vdash z} \quad \dfrac{\dfrac{\dfrac{b \in dom(T)}{T \vdash b} \quad \dfrac{i \in dom(T)}{T \vdash i}}{T \vdash b[i]} \quad \dfrac{j \in dom(T)}{T \vdash j}}{T \vdash b[i] + j}}{T \vdash z[b[i] + j]}}{T \vdash b[i] \times a < z[b[i] + j]}}{T \vdash x[i] = b[i] \times a < z[b[i] + j]}$$

$strMap["result"] = data[intMap[j]] + strMap["key"];$ سوال ۹

$$\frac{T \vdash a : \tau_1[\tau_2] \quad T \vdash i : \tau_2}{T \vdash a[i] : \tau_1} \qquad \frac{T \vdash a : \tau_1[\tau_2] \quad T \vdash i : \tau_2 \quad T \vdash e : \tau_1}{T \vdash a[i] = e : void}$$

فرض می‌کنیم که T گلوبال طوری که تمام متغیرها و توابع (متغیرها در صورت لزوم) کل آن در آن موجود است، قرار می‌گیرند

$$\frac{(intMap, int[int]) \in dom(T)}{intMap : int[int]} \qquad \frac{(j, int) \in dom(T)}{T \vdash j : int}$$

$$\frac{\dfrac{(strMap, int[string]) \in dom(T)}{T \vdash strMap : int[string]} \quad \dfrac{\text{"result" is a String literal}}{T \vdash \text{"result"} : String}}{T \vdash strMap[\text{"result"}] : int}$$

$$\frac{\dfrac{\dfrac{(data, int[int]) \in dom(T)}{T \vdash data : int[int]} \quad T \vdash intMap[j] : int}{T \vdash data[intMap[j]] : int} \quad \dfrac{\dfrac{(strMap, int[string]) \in dom(T)}{T \vdash strMap : int[string]} \quad \dfrac{\text{"key" is a string literal}}{T \vdash \text{"key"} : string}}{T \vdash strMap[\text{"key"}] : int}}{T \vdash data[intMap[j]] + strMap[\text{"key"}] : int}$$

$$\frac{T \vdash strMap[\text{"result"}] : int \qquad T \vdash data[intMap[j]] + strMap[\text{"key"}] : int}{strMap[\text{"result"}] = data[intMap[j]] + strMap[\text{"key"}] : void}$$

سوال ۷

ترم‌ها

تایپ‌ها

$$T ::= Bool \mid Nat \mid T \to T$$

$$t ::= x \mid true \mid \text{if } t \text{ then } t \text{ else } t \mid 0 \mid \text{succ } t \mid \text{pred } t \mid \text{iszero } t \mid \lambda x : T . x \mid t\,t$$

$$\dfrac{\dfrac{\dfrac{\dfrac{(f, Nat \to Nat) \in \Gamma_2}{\Gamma_2 \vdash f : Nat \to Nat} \quad \dfrac{\dfrac{(f, Nat \to Nat) \in \Gamma_2}{\Gamma_2 \vdash f : Nat \to Nat} \quad \dfrac{(x, Nat) \in \Gamma_2}{\Gamma_2 \vdash x : Nat}}{\Gamma_2 \vdash f(x) : Nat}}{\Gamma_2 = \Gamma_1 \oplus \{(x, Nat)\} \vdash f(f(x)) : Nat}}{\Gamma_1 = \Gamma_0 \oplus \{(f, Nat \to Nat)\} \vdash \lambda x : Nat . f(f(x)) : Nat \to Nat}}{\Gamma_0 \vdash \lambda f : Nat \to Nat . \lambda x . f(f(x)) : (Nat \to Nat) \to (Nat \to Nat)}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{(y, Nat) \in \Gamma_3}{\Gamma_3 \vdash y : Nat}}{\Gamma_3 \vdash \text{isZero } y : bool} \quad \dfrac{\dfrac{(y, Nat) \in \Gamma_3}{\Gamma_3 \vdash y : Nat}}{\Gamma_3 \vdash \text{Succ } y : Nat} \quad \dfrac{\dfrac{(y, Nat) \in \Gamma_3}{\Gamma_3 \vdash y : Nat}}{\Gamma_3 \vdash \text{pred } y : Nat}}{\Gamma_3 = \Gamma_0' \oplus \{(y, Nat)\} \vdash \text{if isZero } y \text{ then succ } y \text{ else pred } y : Nat}}{\Gamma_0' \vdash \lambda y : Nat . \text{if isZero } y \text{ then Succ } y \text{ else pred } y : Nat \to Nat}$$

$$\Gamma_0 \vdash (\lambda f : Nat \to Nat . \lambda x : Nat . f(f(x)))(\lambda y : Nat . \text{if isZero } y \text{ then Succ } y \text{ else pred } y) : Nat \to Nat$$

سوال ⑧ a) ساده سازی اول :

$e_1 \rightsquigarrow$ if $\underbrace{\text{true}}_{\text{boolean}}$ then e_1 else e_2

باید تایپ یکسانی باشد

مثلا $e_1, e_2 : T$

درصورتی پس ساده سازی درست است که e_1 و e_2 هر دو یک type داشته باشند یا به عبارتی well-typed باشند.

با این ساده سازی اگر e_2 یک مشکلی داشته باشد، خطای آن در نظر گرفته نخواهد شد. پس این ساده سازی زمانی درست خواهد بود که هر دوی e_1, e_2 well-typed و از یک type باشند

ساده سازی دوم : این یعنی ساده سازی زمانی درست است که e_2 هم تایپ با تایپ x باشد مثلا $x : T$ و $e_2 : T$

در این صورت ضرب تابع به صورت $T \to \underbrace{T_2}_{\text{تایپ } e_1}$ می‌شود.

b) ساده سازی اول : در این حالت چون شرط دو if همواره درست است پس e1 اجرا می شود و یا بالعکس (در شرط ؟؟)
زمانی که e2 دارای عوارض جانبی یا side effect باشد می تواند باعث تغییر رفتار برنامه شود. اگر تایپ های e1 و e2 فرق کند ممکن است
که (همانطور که در a گفته شد) اگر آن را ~~[illegible]~~ حذف آن بهینه کنیم دیگر ممکن است خطای Cast رخ ندهد و از ما خطا نگیرد و کامپایل.
بنابراین زمانی این بهینه سازی درست است که تایپ ها هم نوع باشند و عبارت e2 دارای side effect یا عوارض جانبی نباشد

c) روند درست اینه که اول type check شود و بعد ساده سازی ها. با این کار ارور ها یافته می شود و امکان اصلاح آنها
وجود دارد. ولی اگر اول ساده سازی کنیم کستی ها ایجاد دار ممکن است که حذف بشوند. شاید به روند اجرای برنامه تاثیری نداشته باشد
ولی خب اصولی نیست این کار. حتی بعد از ساده سازی هم اگر مثل قبل آن type check را انجام دهیم می توان مطمئن شد
که بعد از ساده سازی برنامه مشکل تایپی نداریم.