پیش از پاسخ دادن به این مسئله به نکات زیر دقت نمایید.

- این مسئله شامل دو بخش است که ورودی‌های آنها مشابه است اما بخش دوم صرفاً یک محاسبه اضافه دارد. توصیه می‌شود ابتدا حل بخش اول را به طور کامل انجام دهید و در صورتی که زمان داشتید به حل بخش دوم اقدام نمایید. در نهایت شما یک برنامه را به عنوان پاسخ ارسال خواهید کرد، که می‌تواند فقط خروجی بخش اول را تولید کند یا خروجی بخش دوم را نیز داشته باشد.
- به منظور سهولت تصحیح، برنامه‌تان را فقط در یک فایل بنویسید که نام فایل شماره دانشجویی شما و پسوند آن cpp است. این فایل را تا پیش از زمان مقرر در صفحه سی‌ای‌سی‌ام درس در محل مشخص‌شده با عنوان Final Exam – Part 2 [Upload Here] آپلود نمایید.
- **معیارهای ارزیابی برنامه شما پس از شرح صورت مسئله ذکر شده است.**

بخش اول

هدف این مسئله نوشتن یک مفسر (interpreter) برای زبان برنامه‌نویسی ساده (زبرساد) است. مثال زیر یک نمونه کوچک از یک برنامه زبرساد را نشان می‌دهد:

```
۱  ? m
۲  ? n
۳  diff = m - n
۴  num = n + diff
۵  ! num
۶  u = diff + num * n + 1
۷  ! u / 2
```

هر برنامه زبرساد از دنباله‌ای از دستورات تشکیل شده که به طور متوالی اجرا می‌شود. دستورات سه نوع دارند:

دستور ورودی: این دستور قالبی به شکل `? <variable>` دارد و نمونه‌هایی از آن را در خط‌های ۱ و ۲ برنامه فوق ملاحظه می‌نمایید. اجرای این دستور باعث می‌شود مفسر زبرساد اجرای برنامه را متوقف کند تا کاربر یک عدد صحیح در ورودی وارد کند. این مقدار در متغیر <variable> قرار می‌گیرد و اجرای برنامه ادامه می‌یابد.

دستور خروجی: این دستور قالبی به شکل `! <expression>` دارد و نمونه‌هایی از آن را در خط‌های ۵ و ۷ برنامه فوق ملاحظه می‌نمایید. اجرای این دستور باعث می‌شود مفسر زبرساد مقدار عبارت <expression> را در یک خط مجزا در خروجی بنویسد.

دستور جایگزینی: این دستور قالبی به شکل `<variable> = <expression>` دارد که نمونه‌هایی از آن را در خط‌های ۳، ۴ و ۶ ملاحظه می‌نمایید. با اجرای این دستور مقدار عبارت <expression> در متغیر <variable> کپی می‌شود.

عبارت‌ها در زبرساد متشکل از اعداد صحیح (نامنفی) و متغیرها هستند که با چهار عملگر جمع (+)، تفریق (–)، ضرب (*) و خارج قسمت (/) ترکیب می‌شوند. اولویت این عملگرها یکسان هستند و همگی از راست به چپ ارزشیابی می‌شوند. مثلاً عبارت سمت راست عملگر = در سطر ۶ به این صورت ارزشیابی می‌شود: diff + (num * (n + 1)). به این ترتیب، اگر مقادیر ۳ و ۲ به عنوان ورودی (به جای m و n) داده شود، خروجی برنامه به ترتیب ۳ و ۵ خواهد بود. در زبرساد نیازی به تعریف متغیرها نیست و اگر یک متغیر پیش از مقداردهی مورد استفاده قرار بگیرد مقدار آن صفر فرض می‌شود.

نام متغیرها فقط متشکل از حروف بزرگ و کوچک است. فرض کنید اعداد ثابت نامنفی هستند و در تایپ int جا می‌گیرند و محاسبات باعث سرریز[1] نمی‌شوند. در دو طرف متغیرها، اعداد، عملگرها و نویسه‌های =، ? و ! می‌تواند تعداد دلخواهی (صفر یا بیشتر) فاصله خالی باشد. هر دستور در یک خط نوشته می‌شود و خط‌های خالی نادیده گرفته می‌شوند.

برنامه شما یک برنامه زبرساد را به همراه یک ورودی برای آن دریافت می‌کند و خروجی برنامه را به ازای آن ورودی مشخص می‌کند. ورودی برنامه زبرساد با یک خط در انتهای ورودی برنامه شما مشخص می‌شود که در ابتدای آن یک نویسه $ قرار دارد و بعد از آن تعدادی عدد صحیح (به تعداد دستورات ورودی برنامه زبرساد) ذکر شده‌اند که با یک فاصله خالی از یکدیگر جدا شده‌اند. خروجی برنامه شما باید همان خروجی برنامه زبرساد باشد. فرض کنید برنامه زبرساد از نظر نحوی درست است و ورودی‌های آن نیز به درستی مشخص شده‌اند.

یک ورودی و خروجی نمونه برنامه شما در جدول زیر نمایش داده شده است.

ورودی نمونه	خروجی نمونه
?a ? b ! b+c c = a +b * 2 !c /3 $ 2 4	4 3

[1] overflow

بخش دوم

در این بخش فرض کنید بیش از یک پردازنده برای اجرای برنامه زبرساد در اختیار داشته باشیم. در این صورت ممکن است بتوانیم بعضی از دستورات را موازی یکدیگر اجرا کنیم. به عنوان مثال برنامه زیر را در نظر بگیرید که به طور عادی در ۸ قدم اجرا می‌شود.

```
۱  ? a
۲  d = a-6*4
۳  x = 1+a/2
۴  a = 3
۵  y = a+x-1
۶  e = 12*3
۷  z = d*2
۸  ! y+e
```

این برنامه را می‌توان با داشتن دو پردازنده در ۵ قدم اجرا کرد و همان نتیجه را داشت.

```
۱  ? a
۲  d = a-6*4      x = 1+a/2
۳  a = 3          z = d*2
۴  y = a+x-1      e = 12*3
۵  ! y+e
```

در اینجا منظور از «نتیجه برنامه» نه تنها خروجی‌های برنامه بلکه مقادیر تمام متغیرها را نیز شامل می‌شود.

در برنامه فوق چون دستور x=1+a/2 به نتیجه دستور d=a-6*4 وابسته نیست می‌توان آنها را موازی اجرا کرد. دقت نمایید علیرغم این که دستور a=3 به دستورات قبل از خود وابسته نیست اما چون مقدار a را تغییر می‌دهد و دستورات قبلی به a وابسته هستند باید بعد از آنها اجرا شود. توجه کنید که موازی کردن این دستور با دو دستور قبل از خودش نیز مجاز نیست چون ممکن است اجرای آن زودتر از دستورات موازی انجام شود و نتیجه آنها را تحت تأثیر قرار دهد.
در مورد دستورات ورودی نیز دو قاعده وجود دارد:

۱. هیچ‌وقت دو دستور ورودی موازی هم انجام نشوند
۲. ترتیب خواندن ورودی‌ها باید حفظ شود

مثال زیر به‌کارگیری این قواعد را نشان می‌دهد:

قبل از موازی‌سازی	بعد از موازی‌سازی
? a ? b c = a + b ? d ? e ! c + d	? a ? b c = a + b ? d ? e ! c + d

در این قسمت برنامه شما باید بعد از نوشتن خروجی برنامه زبرساد، کم‌ترین تعداد قدم‌های اجرای این برنامه را که نتیجه آن با برنامه ترتیبی (غیرموازی) یکی باشد را به فرمت `Minimum Steps: <steps>` بنویسد. تعداد پردازنده‌ها را نامحدود فرض کنید. یک ورودی و خروجی نمونه این قسمت را در زیر ملاحظه می‌نمایید.

نمونه ورودی	نمونه خروجی
? a d = a-6*4 x = 1+a/2 a = 3 y = a+x-1 e = 12*3 z = d*2 ! y+e $ 1	39 Minimum Steps: 5

ارزیابی برنامه

در ارزیابی برنامه شما معیارهای زیر مدنظر قرار می‌گیرند.

معیارهای ارزیابی برنامه	امتیاز
درستی عملکرد بخش اول	۴۰
درستی عملکرد بخش دوم	۲۰
رعایت اصول طراحی و برنامه‌نویسی شیءگرا • تعریف کلاس‌ها متناسب با صورت مسئله • قراردادن داده‌ها و محاسبات در کلاس‌های مناسب • کپسوله‌سازی[2] داده‌ها • استفاده به‌جا از رابطه‌های وراثت و چندریختی	۲۵
رعایت خوانایی برنامه • نام‌گذاری مناسب • پرهیز از توابع طولانی • رعایت اصول دندانه‌گذاری[3] • نام‌دهی به‌جا به ثابت‌ها	۱۵
مجموع	۱۰۰

سنجش درستی عملکرد برنامه شما توسط تعدادی آزمایه[4] صورت می‌گیرد که هر کدام از آنها امتیاز مجزا دارد. دقت نمایید آزمایه‌ها به تدریج ویژگی‌های برنامه را پوشش می‌دهند. مثلاً اولین آزمایه صرفاً برنامه 3! را می‌آزماید و دومین آزمایه برنامه‌ای را که فقط شامل دو سطر x? و x! است. بنابراین اگر برنامه شما روی همه ورودی‌ها به درستی کار نمی‌کند، سعی کنید دست‌کم ورودی‌های ساده را به درستی پردازش کنید.

[2] encapsulation
[3] indentation
[4] test case