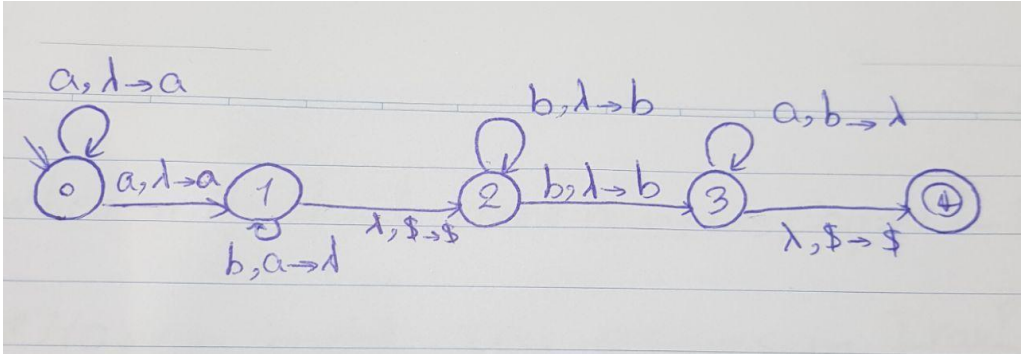
گذارش Q4 :



ورودی : یک استرینگ به عنوان رشته ورودی وارد میکنید.

خروجی: خروجی شامل n خط میباشد. خط i ام شامل راس ابتدا، انتها و یال حرکت i ام میباشد.

نحوه اجرای کد: در ابتدا تابع build اجرا میشود که یال های NPDA مورد نظر اضافه میشد. سپس تابع dfs اجرا میشود که دو ورودی دارد، اولی نشان دهنده استیت فعلی در هر لحظه و دومی نشان دهنده محل فعلی روی رشته ورودی است. در هر بار از اجرای تابع dfs ابتدا چک میشود که اگر رشته ورودی به پایان رسیده و در یک استیت پایانی هستیم یعنی به جواب رسیده‌ایم و true برمیگرداند. در غیر این صورت همه یال های استیت فعلی را بررسی میکند و اگر یالی یکسان با کاراکتر فعلی از رشته ورودی بود (یعنی لیبیل یال برابر استرینگ ورودی باشد و هم کاراکتری که باید از استک حذف شود برابر آخرین خانه استک باشد و یا هر کدام نال(#) باشند) ابتدا استک را آپدیت کرده (پوش و پاپ را در صورت نال(#) نبودن انجام میدهد) و سپس dfs را برای استیت جدید صدا میکند و اگر true برگرداند یعنی یک جواب پیدا شده

و true برمیگرداند و اگر نه استک را دوباره به حالت اولیه برمیگرداند تا برای حرکت های دیگر تغییری نکرده باشد. در آخر اگر یال مناسبی پیدا نشد یعنی جوابی وجود ندارد و false برمیگرداند. در نهایت اگر جوابی پیدا شد حرکات چاپ میشوند.

توضیحات بیشتر: هر ماشین به صورت گراف نگه‌داری میشود، به صورت لیست مجاورت. وکتور adj[i] راس های مجاور استیت i ام و به طور متناظر وکتور w[i] لیبل هر یال خروجی از استیت i ام، وکتور pop[i] مقداری که باید از استک حذف شود و push[i] مقداری که باید به آخر استک اضافه شود را نشان میدهند. آرایه isFinal نشان میدهد که آیا استیت i ام یک استیت پایانی هست یا نه. رشته ورودی در استرینگ s قرار میگیرد. ansState راس هایی که برای رسیدن به جواب باید به ترتیب پیموده شوند را نگه میدارد و ansW ، ansPop و ansPush به ترتیب لیبل، پوش و پاپ های یال مربوط به آن را.