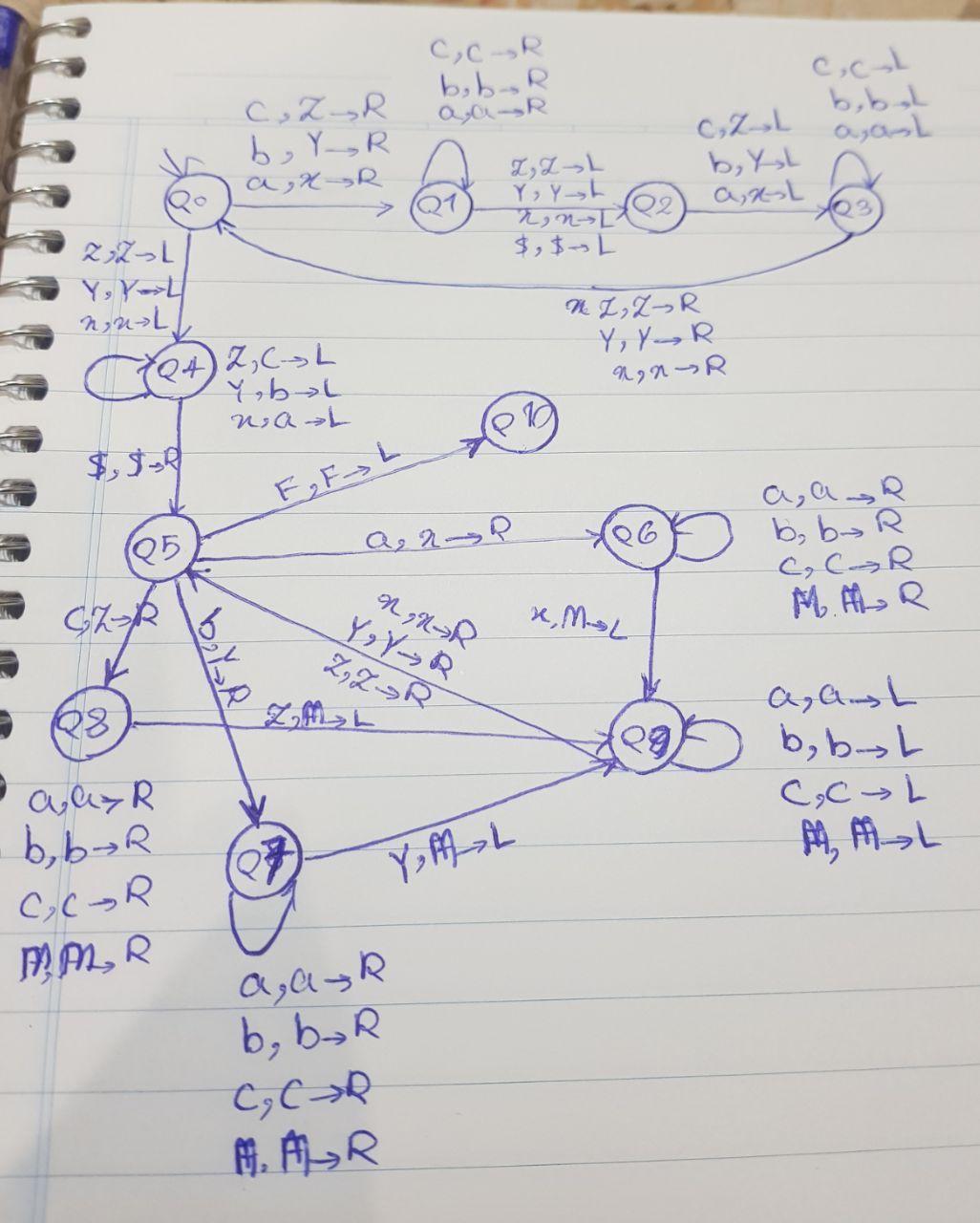
گذارشQ5 :



ورودی : یک استرینگ به عنوان رشته ورودی وارد میکنید.

خروجی: خروجی شامل 2n خط میباشد. خط 2i - 1 ام شامل راس ابتدا، انتها و یال حرکت i ام میباشد و خط 2i برای نشان دادن وضعیت رشته در هر لحظه است.

نحوه اجرای کد: در ابتدا تابع build اجرا میشود که یال های ماشین تورینگ مورد نظر اضافه میشد. از آنجایی که ما رشته بینهایت نداریم و فقط یک کاراکتر بعد و قبل از رشته ورودی ور ماشین تورینگ مورد نظر استفاده میشود، فقط یک $ با ابتدا و انتهای رشته ورودی اضافه میکنیم. 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 (که چون یک $ به ابتدای ورودی اضافه کردیم در ابتدا 1 است). در هر بار از اجرای تابع dfs ابتدا چک میشود که اگر در یک استیت پایانی هستیم یعنی به جواب رسیده‌ایم و true برمیگرداند. در غیر این صورت همه یال های استیت فعلی را برسی میکند و اگر یالی یکسان با کاراکتر فعلی از رشته ورودی بود (کاراکتری که روی آن قرار داریم برابر کاراکتر read یال مورد نظر بود) ابتدا رشته را آپدیت کرده (کاراکتر write را روی مکان فعلی مینویسد) و سپس dfs را برای استیت جدید صدا میکند (و با توجه به move مکان روی رشته را آپدیت میکند) و اگر true برگرداند یعنی یک جواب پیدا شده و true برمیگرداند و اگر نه رشته 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 R[i] کاراکتر قابل خواندن هر یال خروجی از استیت i ام، وکتور W[i] مقداری که باید جایگذین کاراکتر فعلی شود و M[i] مشخص میکند حرکت به سمت چپ است یا راست. آرایه isFinal نشان میدهد که آیا استیت i ام یک استیت پایانی هست یا نه. رشته ورودی در استرینگ input قرار میگیرد. ansState راس هایی که برای رسیدن به جواب باید به ترتیب پیموده شوند را نگه میدارد و ansR ، ansW و ansM به ترتیب کاراکتر قابل خواندن، کاراکتر جایگزین و حرکت های یال مربوط به آن را.