YOUR NAME: REGISTRATION #:

(15 points)

(G) Sk8 Parsr (1/4)

Languages are everywhere… even in places where you don't expect them.
Consider the "combo rules" of *P-Little's Triple-I XTreem Hyp0th3tica7 Sk8boarding Game*. In it, players press a series of buttons (left, right, down, up, circle, triangle, square, and X) to make their on-screen avatar perform skateboard tricks that illustrate pro boarder P-Little's "Triple-I" philosophy of Insane, Ill-Advised, and Impossible According to the Laws of Physics. Underneath, the game is using the methods of computational linguistics to turn this "little language" of button presses into tricks and combos. The game uses a simple ***shift-reduce parser*** to parse button "words" into combo "sentences". As each button-press comes in, the corresponding symbols are placed, in order, in a *buffer* (that is, temporary storage space).

1. ↑
2. ↑ ←
3. ↑ ← ⊡
4. ↑ ← ⊡ ⊗

If, at any point, the *rightmost* symbols in this buffer match any of the patterns on the next page, they are removed and replaced with a new symbol indicating a combo. So, since ⊡⊗ corresponds to an "ollie", we replace it with the new symbol **Ollie**.

5. ↑ ← **Ollie**
6. ↑ ← **Ollie** ⊡
7. ↑ ← **Ollie** ⊡ ⊡
8. ↑ ← **Ollie** ⊡ ⊡ ⊗
9. ↑ ← **Ollie** ⊡ **Ollie**

More complex combos can then be built out of simpler combos. You see in the fifth rule on the next page that **Ollie** and **Nollie** can be joined by ⊡ to make a new combo. There are also ***rule schemas*** that can create new combos out of *any* kind of combo. The tenth rule on the next page says that *any* combo (represented by α), whether it's an Ollie or an Inverted-360-Kickflip, can be joined with itself by a ⊡ to make a Double combo:

10. ↑ ← **Double-Ollie**

YOUR NAME: REGISTRATION #:

(G) Sk8 Parsr (2/4)

The chart of shift-replace rules is given below… but with some holes in it.

If the right side of the input matches…		*… replace it with …*
← ↑ △	⇒	**Backside-180**
____________________	⇒	**Frontside-180**
□ ⊗	⇒	**Ollie**
____________________	⇒	**Nollie**
Nollie □ Ollie	⇒	__________
↓ ↓	⇒	**Crouch**
____________________	⇒	**Backside-360**
____________________	⇒	**360-Kickflip**
____________________	⇒	__________
α □ α	⇒	**Double-α**
Double-α □ α	⇒	**Triple-α**
Double-α □ **Double-α**	⇒	**Quadruple-α**
____________________	⇒	**Atomic-α**

YOUR NAME: REGISTRATION #:

(G) Sk8 Parsr (3/4)

Complex combos can get pretty involved. Here are a few combos from the manual to give you an idea:

Inverted-Nollie: ↓⊗□↑
Double-Inverted-Woolie: ↓⊗□□□⊗↑□↓⊗□□□⊗↑
Inverted-Triple-Backside-180: ↓←↑△□←↑△□←↑△↑
Atomic-Double-Frontside-180: →↓◎□→↓◎□↓→↓◎□→↓◎↑
Inverted-Backside-360: ↓←↑△→↓◎↑
Triple-360-Kickflip: ↓↓←↑△→↓◎□↓↓←↑△→↓◎□↓↓←↑△→↓◎

G1. How would you perform an “Inverted-Atomic-Backside-360”?

G2. How about an “Atomic-Atomic-Ollie”?

YOUR NAME: REGISTRATION #:

(G) Sk8 Parsr (4/4)

G3. The shift-reduce rules given above are incomplete. Using the descriptions of advanced combos in the manual, can you fill in the missing pieces? State them as concisely as possible. Use the space on page 2/4.

G4. During playtesting, the testers discover that even though combos like "Quadruple -Ollie" and "Quadruple-Inverted-Woolie" are listed in the manual, the game can never actually recognize any Quadruple combo that the player performs. Why not? How could you fix the game so that it can?

G5. What other types of combinations of the listed combos can never actually be pulled off by the player, and why not?