# HW10

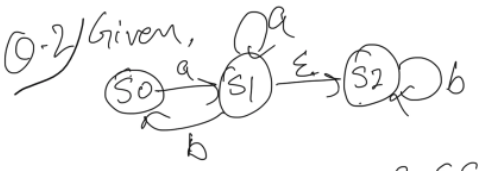
**Q1. Give an equivalent Perl regex for the given FSA.**

```
my $r = qr/\b(ab)*(aa*b*)\b/;
my $my_str = $ARGV[0];

if ($my_str =~ $r) {
    print("Matched \n");
} else {
    print "Not Matched \n";
}
```

```
HW10 ➤ perl p1.pl a
Matched
HW10 ➤ perl p1.pl b
Not matched
HW10 ➤ perl p1.pl aa
Matched
HW10 ➤ perl p1.pl ab
Matched
HW10 ➤ perl p1.pl abba
Not matched
```

**Q2. Convert the NDFSA to a (deterministic) FSA. Draw the machine.**



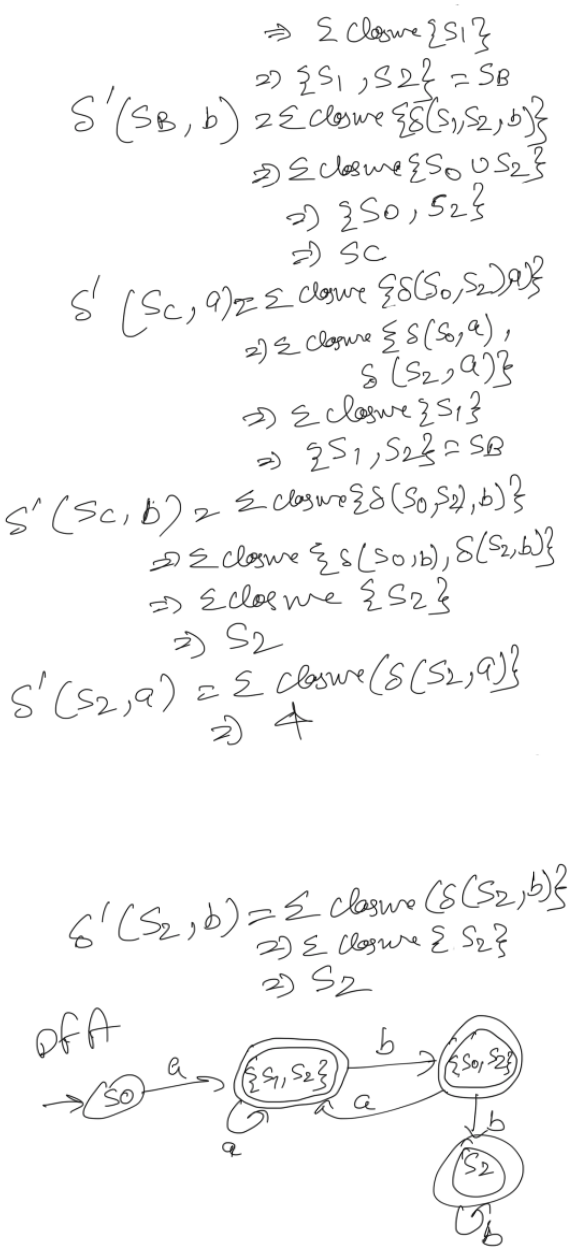

ε closure of {S0} = {S0}
ε closure of {S1} = {S1, S2}
ε closure of {S2} = {S2}

Let ε closure of S0 be state SO

$\delta'(S0, a)$ = ε closure{δ(S0, a)}
= ε closure {S1}
= {S1, S2}
= SB

$\delta'(S0, b)$ = ε closure{δ(S0, b)}
= ε closure {4}
= 4

$\delta'\{SB, a\}$ = ε closure{δ(S1, S2}, a}
= ε closure{δ(S1, a), δ{S2, a)}

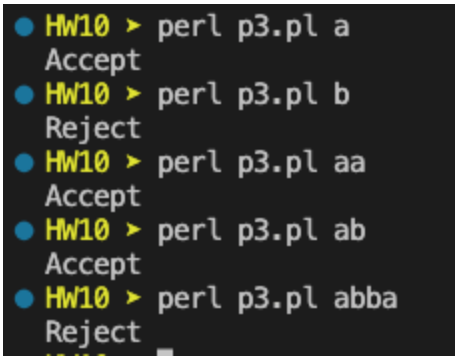



**Q3. Give the implementation of the FSA in Perl.**

```
%delta = (
    s0 => { a => "sb" },
    sb => { a => "sb", b => "sc"},
    sc => { a => "sb", b => "s2"},
    s2 => { b => "s2"}
);
$state = "s0";
foreach $c (split (//, @ARGV[0])) {
    $state = $delta{$state}{$c};
```

```
}
print(($state eq "sb" or $state eq "sc" or $state eq "s2" ) ? "Accept\n" : "Reject\n")
```

**Output:**




**Q4. Run your two Perl programs and give examples:**

- **a, *b, aa, ab, *ba, aaab, abaabb, *abba, *abaabbaaabbb**