Consider the pushdown automaton **A = ( { q, f } , { 0 , 1 , c } , { 0, 1, c, $ }, , q, $, { f } )**, where **** defines transitions:

**( q , 0 , $ ) = { ( q , PUSH(1)) }**

**( q , 1 , $ ) = { ( q , PUSH(0)) }**

**( q , 0 , 0 ) = { ( q , PUSH(1)) }**

**( q , 0 , 1 ) = { ( q , PUSH(1)) }**

**( q , 1 , 0 ) = { ( q , PUSH(0)) }**

**( q , 1 , 1 ) = { ( q , PUSH(0)) }**

**( q , c , 0 ) = { ( p , PUSH(c)) }**

**( q , c , 1 ) = { ( p , PUSH(c)) }**

**( p , λ , c ) = { ( p , POP) }**

**( p , 0 , 0 ) = { ( p , POP) }**

**( p , 1 , 1 ) = { ( p , POP) }**

**( p , λ , $ ) = { ( f , POP) }**

 the language E(**A) = { w c h(w)R | w ∈ {0,1}+ } and h(0)=1; h(1)=0**

Write the equivalent grammar using our class’s variant of the construction in Hopcroft, Motwani and Ullman.

Hint: the starting non-terminal is: **< q , $ , f >**, meaning generate all string that are consumed when we start in **q**, and end up in **f**, having uncovered what’s below **$**.

**< q, $, f > → 0 < q, 1, q > < q, $, f > X**

**| 0 < q, 1, p > < p, $, f >**

**| 0 < q, 1, f > < f, $, f > X**

**| 1 < q, 0, q> < q, $, f > X**

**| 1 < q, 0, p > < p, $, f >**

**| 1 < q, 0, f > < f, $, f > X**

**< q, 0, p > → 0 < q, 1, p > < p, 0, p >**

**| 1 < q, 0, p > < p, 0, p >**

**| c <p, 0, p >**

**< q, 1, p > → 0 < q, 1, p > < p, 1, p >**

**| 1 < q, 0, p > < p, 1, p >**

**| c <p, 1, p >**

**< p, 0, p > → 0**

**< p, 1, p > → 1**

**< p, $, f > → λ**

**< q, 0, q>, < q, 1, q >, < f, $, f >** can lead nowhere as states q and f never entered after popping the stack.

**Rewrite as**

**S → 0 T | 1 U**

**T → 0 T 1 | 1 U 1 | c 1 // owe you a 1**

**U → 0 T 0 | 1 U 0 | c 0 // owe you a 0**

Consider the pushdown automaton **A = ( { q, f } , { 0 , 1 , c } , { 0, 1, c }, , q, { f } )**, where **** defines transitions:

**( q , 0 , λ ) = { ( q , PUSH(1)) }**

**( q , 1 , λ ) = { ( q , PUSH(0)) }**

**( q , c , λ ) = { ( p , PUSH(c)) }**

**( p , λ , c ) = { ( p , POP) }**

**( p , 0 , 0 ) = { ( p , POP), (f, POP) }**

**( p , 1 , 1 ) = { ( p , POP), (f, POP) }**

 the language E(**A) = { w c h(w)R | w ∈ {0,1}+ } and h(0)=1; h(1)=0**

Write the equivalent grammar using book’s construction.

Hint: the starting non-terminal is: **Aq,f**, meaning generate all string that are consumed when we start in **q**, and end up in **f**. Note that the stack is empty at start and we allow top of stack to be ignored in transitions.

**Aq,f → Aq,q Aq,f Useless as becomes Aq,f→Aq,f since can never**

 **pop and end in q**

**| Aq,p Ap,f Useless as can never push starting in p**

**| Aq,f Af,f Useless as becomes Aq,f→Aq,f (see below)**

**| 0 Aq,p 1**

**| 1 Aq,p 0**

**Aq,p → Aq,q Aq,p Useless as becomes Aq,p→Aq,p since can never**

 **pop and end in q**

**| Aq,p Ap,p Useless as becomes Aq,p→Aq,p (see below)**

**| 0 Aq,p 1**

**| 1 Aq,p 0**

**| c Ap,p λ This reduces to Aq,p→c**

**Ap,p → λ No other options as never pushes onto stack**

**Af,f → λ No other options since never pushes onto stack**

**Rewrite as**

**S → 0 T 1 | 1 T 0**

**T → 0 T 1 | 1 T 0 | c**