Wojciech Goralewski s464983

**Zadanie algorytmiczne nr 3**

Na początku sprawdzę czy podane słowo na wejściu nie jest puste a następnie jeżeli walidacja przeszła pomyślnie z podanego słowa na wejściu będę szukał symbolu na pozycjach – od pierwszego do ostatnie i następnie tworzył nowe listy za pomocą MAKELIST(obiekt, lista) i podstawiał pod starą. W ten sposób powinienem otrzymać P^-1.

Wejście – P

Wyjście – reverseP

**REV(P)**

symbol := e %inicjalizacja zmiennej%

reverseP := [] %inicjalizacja pustej listy%

if n == 0 then %jeżeli długość słowa to 0 czyli słowo puste%

return symbol %zwróć symbol pusty%

for i = 1 to LENGHT(P) %przechodzę przez słowo P%

symbol := POS(P,i)

reverseP := MAKELIST(symbol, reverseP) %tworzę nową listę pomocą MAKELIST i % %podstawiam pod starą listę %

return reverseP

**Zadanie algorytmiczne nr 4.**

Na początku sprawdzam czy słowa P lub Q nie są puste. Jeżeli jedno z nich jest puste to zwracam jako wynik drugie. Do konkatenacji użyję rekurencji, która będzie przechowywać pierwszy wyraz słowa w nadrzędnych wywołaniach a w dalszych będzie używany ogon tej funkcji aż do osiągniecia słowa pustego. Następnie będę łączył pierwsze symbole z P do słowa Q.

Wejście – P, Q

Wyjście – złączone PQ

**CON(P,Q)**

symbol := e

ogon := [] %inicjalizacja zmiennych%

rekurencyjnyWynik := []

if LENGTH(P) == 0

return Q %zwróć Q jeżeli P jest puste %

if LENGTH(Q) == 0

return P %zwróć P jeżeli Q jest puste %

symbol := HEAD(P) %podstawiam pierwszy wyraz słowa P %

ogon := TAIL(P) %podstawiam resztę słowa P%

rekurencyjnyWynik := CON(ogon, Q) % rekurencyjnie wywołuję algorytm%

return MAKELIST(symbol, rekurencyjnyWynik) % zwracam połączoną listę z pierwszym symbolem%