**Опис дій транслятора**

Зчитується файл у массив rows ,де перший iндекс -це номер рядка ,а другий позицiя слова у рядку.

За допомогою створенного словника лексем у виглядi бази данних mySQL та функцiй виявлення типiв лексем за кодом для кожного елементу массиву додаються властивостi ,а саме iм’я лексеми ,тип ,довжина ,та необхiднi лексеми пiд час синтасксичного та сематичного аналiзу .

На етапi синтасксичного аналiзу для всiх лексем послiдовно перевiряються на наявнiсть на необхiдному мiсцi записанi на попередньому етапi лексеми.

Далi перевiрка на правильнiсть розташування вiдповiдних рядкiв вiдносно сегментiв.

На останьому етапi виводиться пiдсумкова таблиця ,а також генеруються рядки помилок .

Після завершення аналізу ,у лістинг записується:

Для кожного рядка:

Вихідний рядок

“ERROR” при наявності помилки

**Опис функцій транслятора**

**Processing.php**

start(‘\*.asm’)

Запускає обробку файлу.

Lexical\_analisation(array\_rows)

Запускає лексичний аналіз массиву записаного на попередньому етапі.

Detect\_lexem(lexem)

Запускає перевірку на тип лексеми кожного слово.

syntax\_analisation(count\_of\_rows,array\_rows)

Запускає синтаксичний та сематичний аналіз массиву лексем .

table(count\_of\_rows,array\_rows)

Виводить результат трансляції.

Check\_errors(count\_of\_rows,array\_rows)

Записує у \*.lst результат роботи транслятора

**Create\_Table.php**

Створює базу данних лексем.

**Вхідний файл**

.386

mcrs MACRO

And parm,dd

ENDM

data segment

str db "big ukrainian boss's string"

hex dw 3Eh

bina dd 01101101b

deca dd 21

data ends

cod1 segment

assume cs:cod1,ds:data

Neg bh

Neg ebx

Dec deca

Adc bh, ptr [ebx+6]

Adc ecx,[ebx+6]

Adc bh,bina

Adc ecx,hexa

Cmp es:[ebx+6],ah

Cmp [ebx+6],eax

Cmp deca,ah

Cmp bina,eax

Or eax,bh

Or bh,bl

Or al,edx

Or ecx,edx

Mov eax,bl

Mov eax ,bx

Mov eax,ebx

Mov al,bl

Mov al,bx

Mov al,ebx

met1:

;And [bx+si],8Ah

And [bx+bp+di+5],8AF2h

And [ebx+6],8AF22E4Bh

And hexa,bh

And bina,bx

And deca,ebx

Jc met1

Jmp met2

cod1 ends

cod2 segment

assume cs:cod2,ds:data

met2:

cod2 ends

END

**Output Listing file**

Marchecko KV-51 v11 listing

0 : .386

1 : mcrs MACRO

2 : And parm , dd

Error near line 2:"register/memory" is required at the next position of lexem on line 2 on position 0

"register/memory/const/variable" is missed after lexem on line 2 on position 0

"identifier" is required at the previous position of lexem on line 2 on position 3

"ds" is required before this row by lexem on line 2 on position 3

3 : ENDM

4 : data segment

5 : str db "big ukrainian boss's string"

6 : hex dw 3Eh

7 : bina dd 01101101b

8 : deca dd 21

9 : data ends

10 : cod1 segment

11 : assume cs:cod1 , ds:data

12 : Neg bh

13 : Neg ebx

14 : Dec deca

Error near line 14:"register/memory" is required at the next position of lexem on line 14 on position 0

15 : Adc bh , ptr [ebx+6]

16 : Adc ecx , [ebx+6]

17 : Adc bh , bina

18 : Adc ecx , hexa

Error near line 18:"register/memory/const/variable" is missed after lexem on line 18 on position 0

19 : Cmp es:[ebx+6] , ah

20 : Cmp [ebx+6] , eax

21 : Cmp deca , ah

22 : Cmp bina , eax

23 : Or eax , bh

Error near line 23:Operand types must match on line 23

24 : Or bh , bl

25 : Or al , edx

Error near line 25:Operand types must match on line 25

26 : Or ecx , edx

27 : Mov eax , bl

Error near line 27:Operand types must match on line 27

28 : Mov eax , bx

Error near line 28:Operand types must match on line 28

29 : Mov eax , ebx

30 : Mov al , bl

31 : Mov al , bx

Error near line 31:Operand types must match on line 31

32 : Mov al , ebx

Error near line 32:Operand types must match on line 32

33 : met1:

34 : ;And [bx+si] , 8Ah

35 : And [bx+bp+di+5] , 8AF2h

36 : And [ebx+6] , 8AF22E4Bh

37 : And hexa , bh

Error near line 37:"register/memory" is required at the next position of lexem on line 37 on position 0

38 : And bina , bx

Error near line 38:"register/memory" is required at the next position of lexem on line 38 on position 0

39 : And deca , ebx

Error near line 39:"register/memory" is required at the next position of lexem on line 39 on position 0

40 : Jc met1

41 : Jmp met2

42 : cod1 ends

43 : cod2 segment

44 : assume cs:cod2 , ds:data

45 : met2:

46 : cod2 ends

47 : END

Find 13 errors

**Алгоритм роботи програми**

1) Відбувається читання назв файлів вихідного коду та лістингу з параметрів запуску (в разі наявності) або з клавіатури. В разі необходності, шлях до файлу лістингу генерується автоматично з назви файлу вихідного коду шляхом заміни розширення.

2) Відкриття файлу вихідного коду.

3) Порядкове читання коду.

3.а) Лексичний аналіз прочитаного рядка, формування таблиці лексем. Відсікаються коментарі. Виконується розбиття на лексеми з використанням регулярних виразів. Визначається тип лексеми (за допомогою регулярних виразів (текстові чи числові константи, односимвольні лексеми) та шляхом пошуку в таблиці зарезервованих слів). Лексеми, що не співпали з переліченими типами, перевіряються на максимальну довжину, ті, що її не перевищуюють - користувацькі.

3.б) Формування структури рядка. Типи лексем (для інструкцій - ім'я) копіюється в новий рядок, виконуються заміни для спрощення обробки (загальний вигляд виразів для пам'яті, заміна абсолютного виразу константою).

3.в) Формування зміщень в сегменті. На цьому етапі відбувається порівняння одержаного на етапі 3.б рядка з допустимими за варіантом конструкціями. Заповнюються таблиці сегментів і користувацьких змінних та бібліотека макророзширень. Відповідно до правила (інструкція + операнди) вираховується зміщення команди в сегменті).

4) Заново відкривається файл вихідного коду. Стандартний потік виводу зв'язується з файлом лістингу (перенаправлення виводу).

5) Порядкове читання коду.

5.а) Лексичний аналіз, формування таблиці лексем.

5.б) Формування структури рядка.

5.в) Генерація коду команди. Отриманий на попередньому етапі рядок порівнюється з допустимими варінтами, аналізуються операнди інструкціїї, відбувається формування кодів команд, байтів modr/m, sib, безпосередніх даних. В разі необхідності відбувається вирахування значення абсолютного виразу (за доп. алгоритму Дейкстра). В разі наявності макросу відбувається підстановка його фактичного параметра (за наявності) та читання нового рядку не з файлу вихідного коду, а з бібліотеки макророзширень. (Пункти 5-5.г виконуються для нового рядка). На цьому ж етапі відбувається виявлення помилок (неспівпадіння з правилами, неспівпадіння операндів, перевищення макс.допустимого значення, наявність в рядку лексеми з помилковим типом).

5.г) Отриманий на попередньому етапі код команди виводиться в файл лістингу. В разі наявності в рядку помилки збільшується лічильних помилок.

6) Виведення таблиць сегментів та користувацьких змінних, інформаціїї про макроси та кількості помилок в файл лістингу. Виводиться значення лічильника помилок.

7) Закриття файлів, кінець роботи програми.