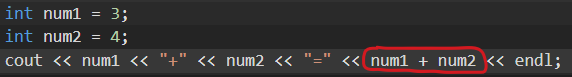
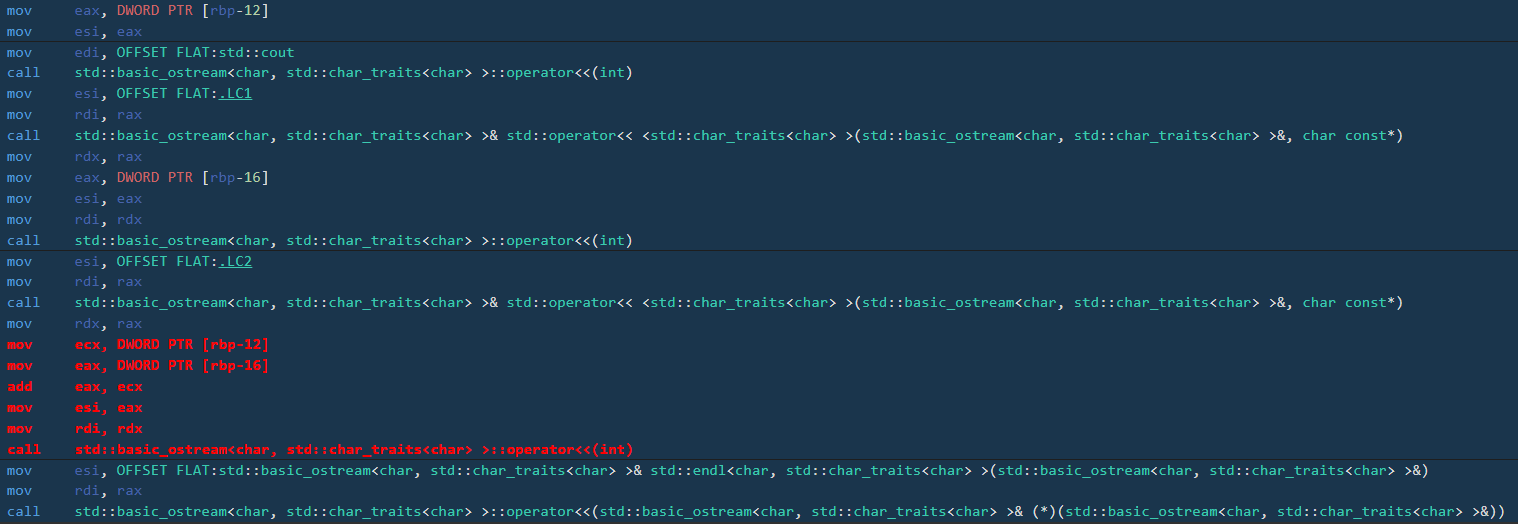
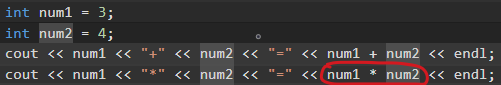
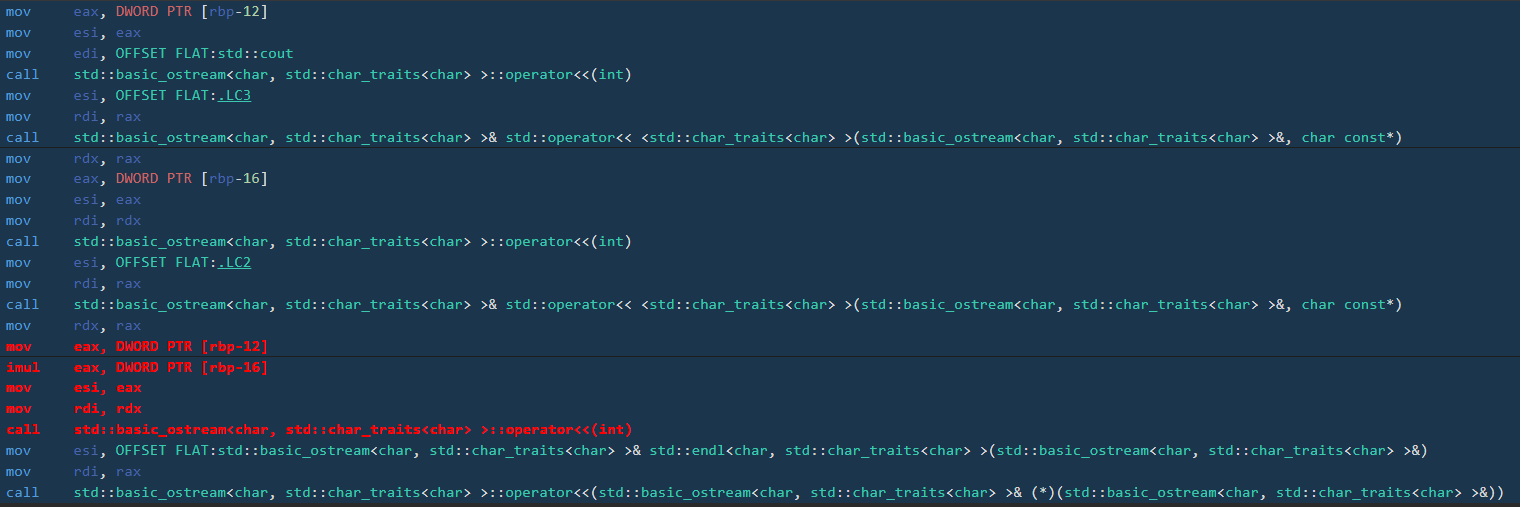
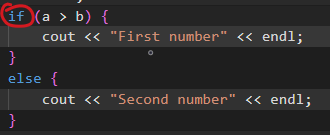
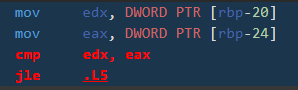
Лабораторна робота 1 (АКС)

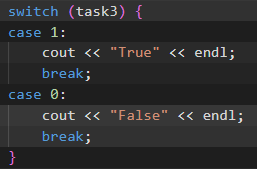
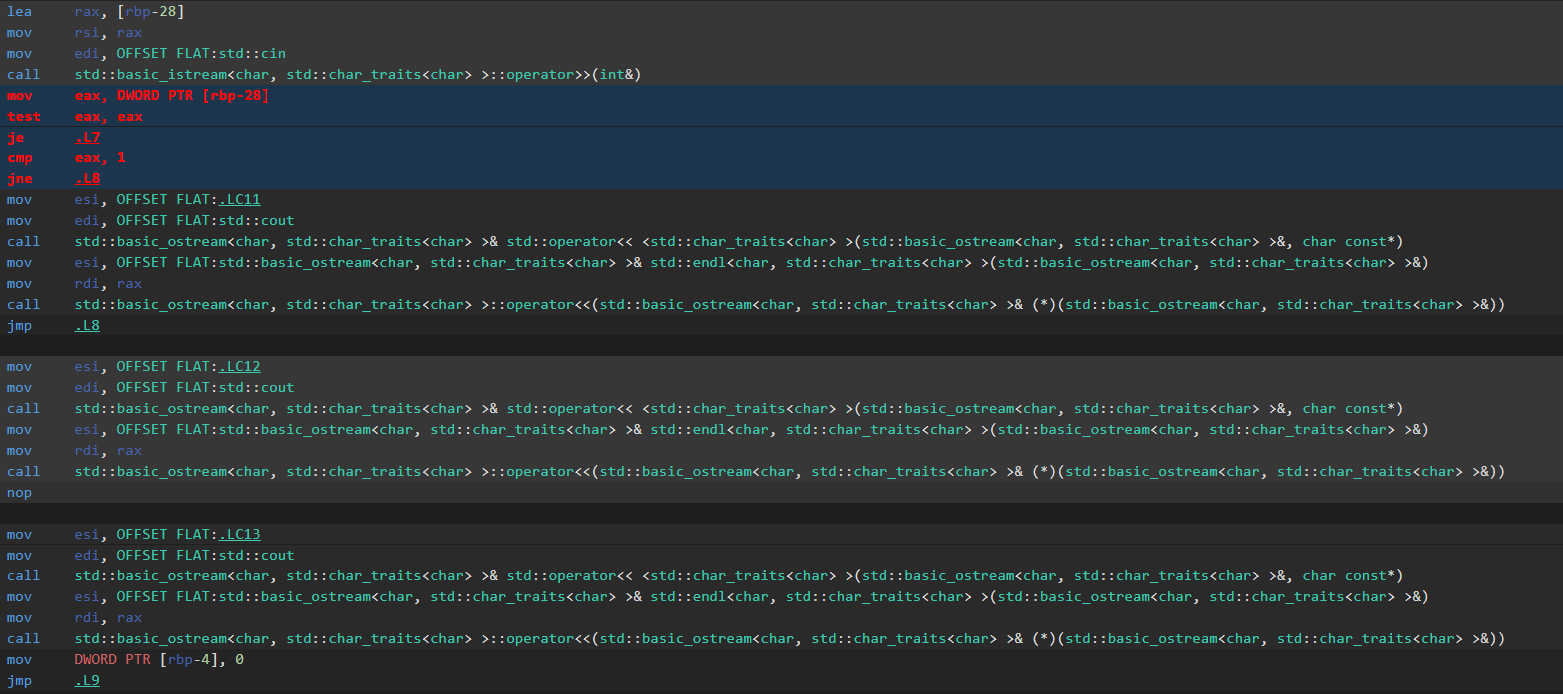
1.1.1. Базові функції додавання в С++ вигляді:  


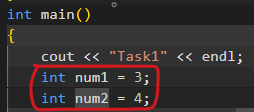
1.1.2. Базові функції додавання в ассемблерному вигляді:  


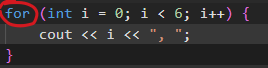
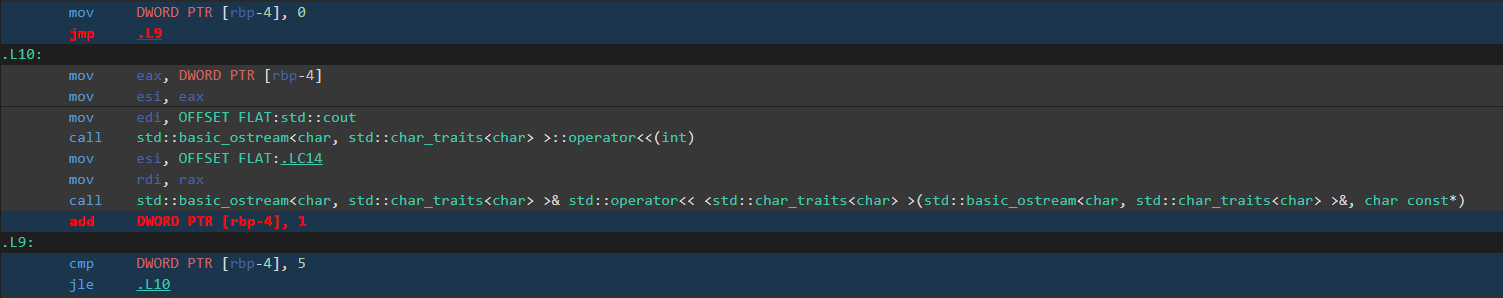
1.2. Базові функції множення в С++ вигляді:  


1.2.1. Базові функції множення в ассемблерному вигляді:  


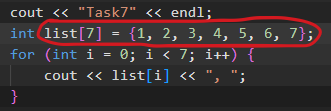
2.1. if стейтмент для чисел в С++:  
  
2.2. if стейтмент для чисел в асемблерному вигляді:  


3.1. Switch на 2 пункти в С++:  
  
3.2. Switch на 2 пункти в асемблерному вигляді:  


4.1. Локальні змінні в С++:  
  
4.2. Локальні змінні в асемблерному вигляді::  


5.1. Цикл for в С++:  
  
5.2. Цикл for в асемблерному вигляді:  


6.1. Виклик Функції в С++:  
  
6.1. Виклик Функції в асемблерному вигляді:  


7.1. Масив з елементів в С++:  
  
7.2. Масив з елементів в асемблерному вигляді:  


Контрольні запитання:

[1. Які команди асемблера відповідають за додавання та віднімання ?](https://studfile.net/preview/5157445/page:36/)  
ADD, SUB

[2. Які команди асемблера відповідають за множення та ділення ?](https://studfile.net/preview/5157445/page:36/)  
MUL, IMUL, DIV, IDIV

3. Як зображаються захардкожені значення змінних у мові асемблер ?  
mov dword ptr [num1], 3

4. Як виглядає if структура у мові асемблер ?  
CMP i JLE, cmp відповідає за поріняння, а jle за перехід з виконанням умови, у if умова що більше, тому, наприклад, якщо а > b іде перехід на наступну частину коду

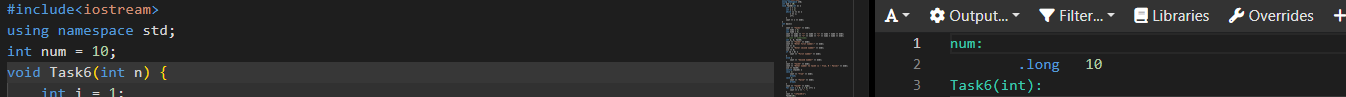
5. Як працює умовний перехід та відповідні оператори ?  
JCC(JLE) – умовний перехід, JZ ADDR – (перевіряється умова чи регістр ZF=1) то виконується перехід на іншу чатину коду. JZ взяли просто до прикладу одного з багатьох умовних переходів

6. Як зображається структура switch у мові асемблер ?  
CMP і JE, (TEST якщо є 0)

7. В чому різниця між глобальними та локальними змінними ?  
Змінні які оголошені поза функцією, але доступні всередині функції(виконуваної операції) називають глобальними. Змінні, які оголошені всередині функції(виконуваної операції), називають локальними. Ці змінні стають недоступними після виходу з функції(виконуваної операції).

Глобальними називаються змінні, які оголошені поза блоком. Вони мають статичну тривалість життя, тобто створюються при запуску програми і знищуються при її завершенні. Глобальні змінні мають глобальну область видимості (або “файлову область видимості”), тобто їх можна використовувати в будь-якому місці файлу, в якому вони оголошені. Локальні відповідно досткпні тільки у блоці

8. Як різниця між глобальними та локальними змінними відображена у мові асемблер ?

Ініціалізація глобальних змінних  


Ініціалізація локальних змінних  


9. В чому різниця між циклами for та while у мові асемблері ?  
Різниці на мові асемблер у зовнішньому вигляді практично немає

10. Як викликається функція у мові асемблер ?  
Через CALL

11. Як викликається системна функція у мові асемблер ?  
Також через CALL

12. В чому особливість створення массиву у мові ассмеблер ?  
У тому, що кожен елемент задається окремо як змінна, яка лежить дуже близько у пам’яті і шістнадцятковий код відрізняється на 4 (у нашому видпаку, бо ми вибрали тип вирівнювання DWORD – сегмент починається з адреси, яка є кратна 4.)

Зачення операцій в асемблері

JLE

Перевіряє стан одного або кількох прапорів стану в регістрі EFLAGS (CF, OF, PF, SF та ZF) і, якщо прапори перебувають у зазначеному стані (умові), виконує перехід до цільової інструкції, визначеної операнд призначення. Код умови (cc) пов’язаний із кожною інструкцією, щоб вказати стан, який перевіряється. Якщо умова не виконується, стрибок не виконується, і виконання продовжується з інструкцією, наступною за інструкцією Jcc.

MOV

Копіює другий операнд (операнд-джерело) у перший операнд (операнд-приймач). Операнд-джерело може бути негайним значенням, регістром загального призначення, сегментним регістром або місцем пам'яті; регістр призначення може бути регістром загального призначення, сегментним регістром або місцем пам'яті. Обидва операнди повинні мати однаковий розмір, який може бути байтом, словом, подвійним словом або чотирисловом.

CMP

Порівнює перший операнд-джерело з другим операндом-джерелом і встановлює позначки стану в регістрі EFLAGS відповідно до результатів. Порівняння виконується шляхом віднімання другого операнда з першого операнда, а потім встановлення позначок стану таким же чином, як інструкція SUB. Коли негайне значення використовується як операнд, воно розширюється за знаком до довжини першого операнда.

SUB

Віднімає другий операнд (операнд-джерело) від першого операнда (операнд-одержувач) і зберігає результат в операнді-одержувачі. Операнд призначення може бути регістром або місцем пам'яті; вихідний операнд може бути негайним, регістром або місцем пам'яті. (Однак два операнди пам’яті не можуть використовуватися в одній інструкції.) Коли як операнд використовується безпосереднє значення, воно розширюється за знаком до довжини формату операнда призначення.

TEST

Обчислює побітове логічне AND першого операнда (операнд джерела 1) і другого операнда (операнд джерела 2) і встановлює прапорці стану SF, ZF і PF відповідно до результату. Потім результат відкидається.