**Кафедра «Информатика и программное обеспечение»**

**ЛАБОРАТОРНАЯ РАБОТА № 0**

**ПО ДИСЦИПЛИНЕ**

**«Архитектура вычислительных систем»**

Выполнил студент гр. О-21-ИВТ-1-по-Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Шпаков В.В.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_ к.ф.-м.н., доц. Дмитроченко О.Н.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

**БРЯНСК 2023**

**Содержание**

[Задание №1 3](#_Toc155652767)

[Задание №2 7](#_Toc155652768)

[Задание №3 8](#_Toc155652769)

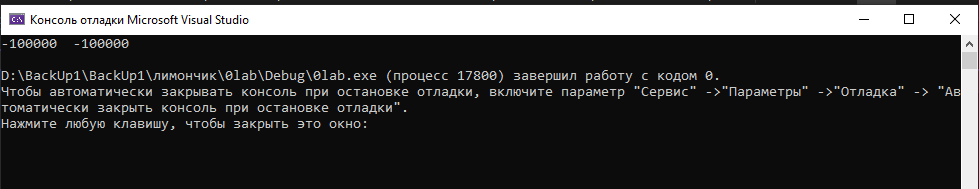
[Заключение 10](#_Toc155652770)

# Задание №1

Пример работы ассемблеровской вставки с вычитанием

*Листинг 1*

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  int a = 200000, b = 300000, s, S;  s = a - b;  \_\_asm {  mov eax, a  sub eax, b  mov S, eax  }  \_\_asm {  jo OVRFLW  }  cout << s << " " << S << endl;  goto CONT;  OVRFLW:  cout << "overflow" << endl;  CONT:  return 0;  } |

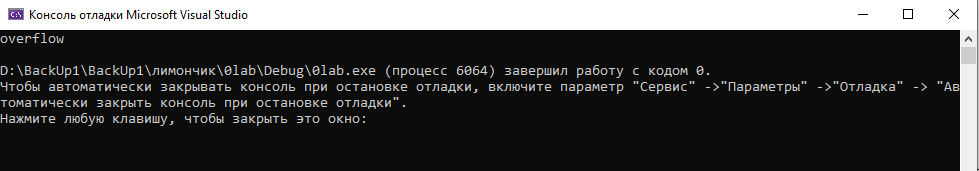
В результате получаем два ответа (см. рис.1)  


*Рис.1 Вывод результата*

*Листинг 2*

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  int a = 3000000000, b = a, s, S;  s = a + b;  \_\_asm {  mov eax, a  add eax, b  mov S, eax  }  \_\_asm {  jo OVRFLW  }  cout << s << " " << S << endl;  goto CONT;  OVRFLW:  cout << "overflow" << endl;  CONT:  return 0;  } |

Тут для вызова «overflow»(см. рис.2) мне пришлось использовать сумму и присвоить переменной a = 30000000000 (см. Листинг 2).



*Рис.2 Вывод результата*

Ставим точку останова как показано на рис.3. Правой кнопкой мыши выбираем переход к дизассемлированному коду (см. рис.4). Результат перехода можно увидеть на рис. 5).

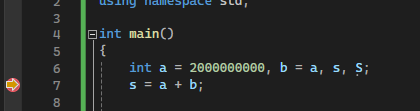


Рис. 3. Точка останова

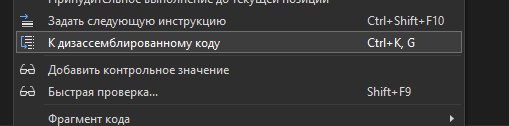


Рис. 4. Переход к дизассемблированому коду

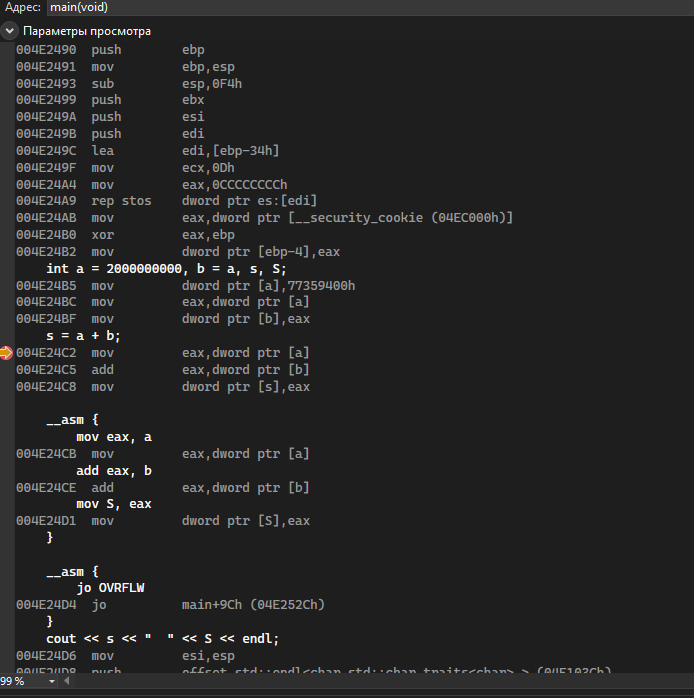
**

Рис. 5. Дизассемлированный код

Пробуем то же самое с другими типами данных. На Листинге 3 можем увидеть short int a = 2000000000 и результат на рис. 6.

*Листинг 3*

#include <iostream>

using namespace std;

int main()

{

short int a = 3000000000, b = a, s, S;

s = a + b;

\_\_asm {

jo OVRFLW

}

cout << s << endl;

goto CONT;

OVRFLW:

cout << "overflow" << endl;

CONT:

return 0;

}

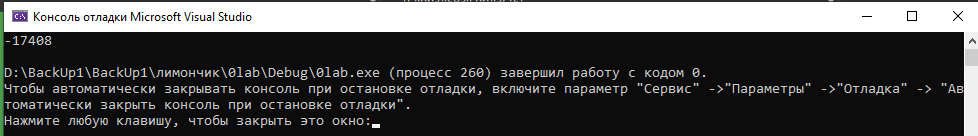
**

Рис. 6. Вывод результата

На Листинге 4 можно увидеть long int a = 2000000000 и результата на рис.7.

*Листинг 4*

#include <iostream>

using namespace std;

int main()

{

long int a = 3000000000, b = a, s, S;

s = a + b;

\_\_asm {

mov eax,a

add eax,b

mov S,eax

}

\_\_asm {

jo OVRFLW

}

cout << s << " " << S << endl;

goto CONT;

OVRFLW:

cout << "overflow" << endl;

CONT:

return 0;

}

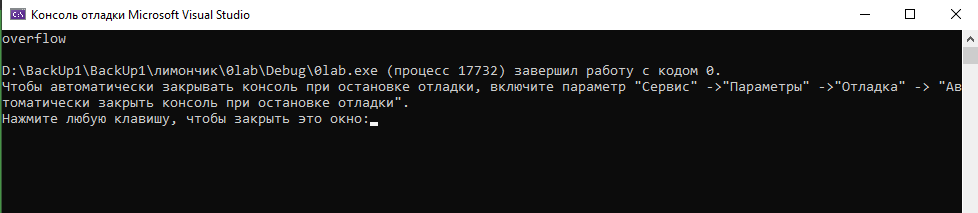
**

Рис. 7. Вывод результата

# Задание №2

На Листинге 5 представлены следующие переменные: double A = 3.3, B = 7.8, C, CC и результат вывода можно увидеть на рис.8.

*Листинг 5*

#include <iostream>

using namespace std;

int main()

{

double A = 3.3, B = 7.8, C, CC;

C = A + B;

\_\_asm {

fld A

fadd B

fstp CC

}

cout << C << " " << CC << endl;

return 0;

}

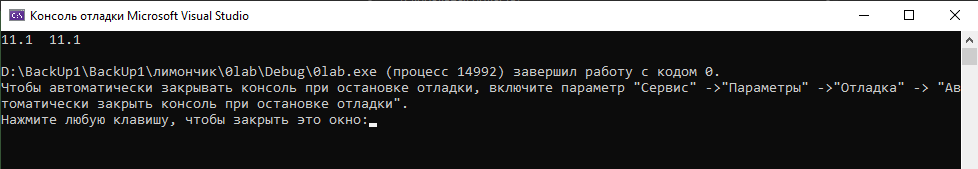
**

Рис. 8.Вывод результата

# Задание №3

Необходимо решить квадратное уравнение. На Листинге 6 представлен код, который позволяет это сделать.

*Листинг 6*

#include <iostream>

using namespace std;

double quadr(double a, double b, double c)

{

double D = b \* b - 4 \* a \* c;

return (-b + sqrt(D)) / 2 / a;

}

double Quadr(double a, double b, double c)

{

double \_2 = 2, \_4 = 4;

\_\_asm {

fld b

fmul b

fld \_4

fmul a

fmul c

fsub

fsqrt

fsub b

fdiv \_2

fdiv a

}

}

int main()

{

double a = 1, b = -3, c = 1, x, X;

x = quadr(a, b, c);

X = Quadr(a, b, c);

cout << x << " " << X << endl;

}

Функция quadr использует стандартный метод для решения квадратного уравнения, в то время как функция Quadr применяет ассемблерный код для решения того же уравнения. Как видно на рисунке 9, результаты, полученные с использованием обеих функций, идентичны.

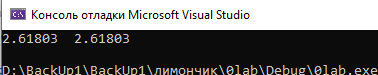


Рис. 9 Вывод результата

# Заключение

В этой лабораторной работе мы познакомились с дизассемблированным кодом, посмотрели результаты с использованием разных типов данных и научились решать квадратное уравнение с помощью ассемблированного кода в Visual Studio.