

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГНС | | 09.00.00 | Информатика и вычислительная техника | | |
| Направление подготовки | | 09.03.01 | Информатика и вычислительная техника | | |
| Направленность (профиль) | |  | Системы автоматизированного проектирования | | |
| Форма обучения | |  | очная | | |
|  | |  |  | | |
| Факультет | |  | Информационных технологий и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Вычислительные системы, сети и телекоммуникации | | |
| Курс | II | | | Группа | 423 |

Отчёт по контрольной работе №1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 423 |  |  |  | Ефремов Иван Андреевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверили: |  |  |  | Макарук Роман Валерьевич |
|  |  | (дата, подпись) |  | Федин Алексей Константинович |
|  |  |  |  |  |

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc158160395)

[2 Исходные данные 3](#_Toc158160396)

[3 Текст учебной программы 3](#_Toc158160397)

[4 Выводы по заданию 9](#_Toc158160398)

## 1 Постановка задачи

Разработать asm-вставку для учебной программы для подсчета суммы, разности, произведения, целой части частного и остатка от целочисленного деления двух целочисленных значений.

## 2 Исходные данные

Учебная программа – подсчет суммы двух целочисленных значений:

#include <iostream>

using namespace std;

int main() {

setlocale(LC\_CTYPE, "Russian");

int firstOperand = 0, secondOperand = 0, sum = 0;

cout << "Введите первый операнд: ";

cin >> firstOperand;

cout << "Введите второй операнд: ";

cin >> secondOperand;

\_asm {

MOV EAX, firstOperand

MOV EBX, secondOperand

ADD EAX, EBX

MOV sum, EAX

}

cout << firstOperand << " + " << secondOperand << " = " << sum << endl;

system("pause");

}

## 3 Текст учебной программы

Исходный код программного решения:

#include <iostream>

#include <string>

using namespace std;

enum options{add = 1, sub, mul, divis, res, quit};

class Operations {

public:

static int Addition(int first, int second) {

int res = 0;

\_asm {

MOV EAX, first

MOV EBX, second

ADD EAX, EBX

MOV res, EAX

}

return res;

}

static int Subtraction(int first, int second) {

int res = 0;

\_asm {

MOV EAX, first

MOV EBX, second

SUB EAX, EBX

MOV res, EAX

}

return res;

}

static int Multiplication(int first, int second) {

int res = 0;

\_asm {

MOV EAX, first

MOV EBX, second

IMUL EAX, EBX

MOV res, EAX

}

return res;

}

static int Division(int first, int second) {

int res = 0;

if (second == 0) {

cout << "\nError! Division by zero!\n\n";

return INT\_MAX;

}

\_asm {

MOV EAX, first

CDQ

MOV EBX, second

IDIV EBX

MOV res, EAX

}

return res;

}

static int Remainder(int first, int second) {

int res = 0;

if (second == 0) {

cout << "\nError! Division by zero!\n\n";

return INT\_MAX;

}

\_asm {

MOV EAX, first

CDQ

MOV EBX, second

IDIV EBX

MOV res, EDX

}

return res;

}

};

void Greeting() {

cout << "Lab #1\n";

cout << "Group 423\n";

cout << "Efremov Ivan Andreevich\n";

cout << "Implemet arithmetic operators using assembler\n";

cout << '\n';

}

void Options() {

cout << "[1] - Addition\n";

cout << "[2] - Subtraction\n";

cout << "[3] - Multiplication\n";

cout << "[4] - Divivsion\n";

cout << "[5] - Remainder\n";

cout << "[6] - Exit\n\n";

}

int ReadInt(string s) {

cout << s;

int tmp = 0;

while (true) {

cin >> tmp;

if (cin.fail()) {

cin.clear();

cin.ignore(INT\_MAX, '\n');

cout << "\nUse correct values!\n\n";

cout << s;

}

else {

cin.clear();

cin.ignore(INT\_MAX, '\n');

return tmp;

}

}

}

int main()

{

Greeting();

int result = 0;

int first = 0, second = 0;

int option = 0;

char sign = '\n';

while (option != options::quit) {

Options();

option = ReadInt(">>");

if ((option >= options::add && option <= options::res)) {

cout << "\nEnter first operand\n";

first = ReadInt(">>");

cout << "\nEnter second operand\n";

second = ReadInt(">>");

}

switch (option) {

case options::add:

result = Operations::Addition(first, second);

sign = '+';

break;

case options::sub:

result = Operations::Subtraction(first, second);

sign = '-';

break;

case options::mul:

result = Operations::Multiplication(first, second);

sign = '\*';

break;

case options::divis:

result = Operations::Division(first, second);

sign = ':';

break;

case options::res:

result = Operations::Remainder(first, second);

sign = '%';

break;

case options::quit:

cout << "\nProgramm finished it's work!\n";

exit(EXIT\_SUCCESS);

default:

cout << "\nThere is no such option in the menu!\n\n";

continue;

}

if (result == INT\_MAX) continue;

cout << '\n' << first << " " << sign << " " << second << " = " << result << "\n\n";

}}

Далее представлены фрагменты ассемблер кода для каждой арифметической операции.

Таблица 1 – Сложение

|  |  |
| --- | --- |
| **Команда ассемблера** | **Комментарий** |
| MOV EAX, first | Помещаем первый операнд в регистр EAX |
| MOV EBX, second | Помещаем второй операнд в регистр EBX. |
| ADD EAX, EBX | Операция сложения аккумулятора и регистра EBX |
| MOV res, EAX | Помещаем результат деления в переменную res. |

Таблица 2 – Вычитание

|  |  |
| --- | --- |
| **Команда ассемблера** | **Комментарий** |
| MOV EAX, first | Помещаем первый операнд в регистр EAX |
| MOV EBX, second | Помещаем второй операнд в регистр EBX. |
| SUB EAX, EBX | Операция вычитания аккумулятора и регистра EBX |
| MOV res, EAX | Помещаем результат деления в переменную res. |

Таблица 3 – умножение

|  |  |
| --- | --- |
| **Команда ассемблера** | **Комментарий** |
| MOV EAX, first | Помещаем первый операнд в регистр EAX |
| MOV EBX, second | Помещаем второй операнд в регистр EBX. |
| IMUL EAX, EBX | Операция умножения аккумулятора и регистра EBX |
| MOV res, EAX | Помещаем результат деления в переменную res. |

Таблица 4 – Деление

|  |  |
| --- | --- |
| **Команда ассемблера** | **Комментарий** |
| MOV EAX, first | Помещаем первый операнд в регистр EAX |
| CDQ | Расширяем знаковый бит содержимого регистра EAX на регистр EDX, чтобы подготовиться к делению с дополнением до двойного слова (64-бита) при делении 32-битных чисел со знаком |
| MOV EBX, second | Помещаем второй операнд в регистр EBX. |
| IDIV EBX | Выполняем деление значения в EAX на значение в EBX.  Результат будет в EAX, остаток от деления - в EDX |
| MOV res, EAX | Помещаем результат деления в переменную res. |

Таблица 5 – Поиск остатка

|  |  |
| --- | --- |
| **Команда ассемблера** | **Комментарий** |
| MOV EAX, first | Помещаем первый операнд в регистр EAX |
| CDQ | Расширяем знаковый бит содержимого регистра EAX на регистр EDX, чтобы подготовиться к делению с дополнением до двойного слова (64-бита) при делении 32-битных чисел со знаком |
| MOV EBX, second | Помещаем второй операнд в регистр EBX. |
| IDIV EBX | Выполняем деление значения в EAX на значение в EBX.  Результат будет в EAX, остаток от деления - в EDX |
| MOV res, EDX | Помещаем остаток в переменную res. |

## 4 Выводы по заданию

В ходе выполнения задания были изучены методы интеграции asm-вставок в код проекта. Были получены практические навыки на примере операций сложения, вычитания, умножения, деления и поиска остатка.