Использование встроенного ассемблера в проектах Microsoft Visual Studio C++ x86

1 Вычисление простейших арифметических выражений

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");
}
```

Задача – разработать asm-вставку для подсчета суммы, разности, произведения, целой части частного и остатка от целочисленного деления двух целочисленных значений.

2 Использование команд сравнения и условного перехода

Учебная программа – определение кратности положительного целого числа пяти:

```
#include <iostream>

using namespace std;

int main() {
  setlocale(LC_CTYPE, "Russian");

  unsigned int number = 0;
  int isNumberDivisibleByFive = 0;
  const int five = 5;

  cout << "N = ";
  cin >> number;

  _asm {
    XOR EDX, EDX
    MOV EAX, number
    DIV five
    CMP EDX, 0
    JNE L01
    MOV isNumberDivisibleByFive, 1
  L01:
  }

  cout << number;
  if (isNumberDivisibleByFive) {
    cout << " - делится на 5 без остатка" << endl;
  }
  else {
    cout << " - делится на 5 с остатком" << endl;
  }

  system("pause");
}
```

Задача – разработать asm-вставку для проверки простоты числа.

Использование внешних подпрограмм в проектах Microsoft Visual Studio C++ x86

Добавляем в проект asm-файл (имя не должно совпадать с cpp-файлом).
Правый клик на проекте, Build Dependencies – Build Customizations.
Выбираем masm и нажимаем ОК.
Правый клик на asm-файле, выбираем Properties.
В поле ItemType выбираем Microsoft Macro Assembler и нажимаем ОК.

3 Команды управления циклом и работы с массивами

Учебная программа – замена каждого отрицательного элемента в динамическом одномерном целочисленном массиве его квадратом и подсчет количества таких замен:

```
// Файл .cpp
#include <iostream>

using namespace std;

extern "C" int MODIFY_ARRAY(int *, int);

int main() {
  setlocale(LC_CTYPE, "Russian");
  int *initialArray = nullptr;
  int i = 0, numberOfReplaces = 0, numberOfElements = 0;
  cout << "Количество элементов массива: ";
  cin >> numberOfElements;
  initialArray = new int[numberOfElements];

  for (i = 0; i < numberOfElements; i++) {
    cout << "[" << i + 1 << "] = ";
    cin >> initialArray[i];
  }

  cout << "Было: ";
  for (i = 0; i < numberOfElements; i++) {
    cout << initialArray[i] << " ";
  }
  cout << endl;

  numberOfReplaces = MODIFY_ARRAY(initialArray, numberOfElements);
```

```
  cout << "Стало: ";
  for (i = 0; i < numberOfElements; i++) {
    cout << initialArray[i] << " ";
  }
  cout << endl;

  cout << "Количество замен: " << numberOfReplaces << endl;

  free(initialArray);
  initialArray = nullptr;
  system("pause");
}
```

```
; Файл .asm
.686
.model FLAT, C
.data
K DWORD 0
.code
MODIFY_ARRAY PROC C X: DWORD, N: DWORD
  type_X = TYPE X
  MOV EBX, X
  MOV ECX, [N]
L01:
  MOV EAX, [EBX]
  CMP EAX, 0
  JGE L02
  MUL EAX
  INC K
  MOV [EBX], EAX
L02 :
  ADD EBX, type_X
  LOOP L01
  MOV EAX, K
  RET
  MODIFY_ARRAY ENDP
END
```

Задача – в учебной программе поменять ручной ввод на заполнение элементов массива случайными целыми значениями в диапазоне от -50 до 50 с использованием внешней asm-подпрограммы, содержащей вызовы функций *time*, *srand*, *rand*.

Просмотр кода дизассемблирования в отладчике Microsoft Visual Studio C++ x86

4 Решение задачи обратной разработки

Учебная программа – часть дизассемблированного кода, полученная с помощью компилятора MSVC x86:

```
push ebp
mov ebp,esp
mov dword ptr [ebp-8],0
mov eax,dword ptr [ebp+8]
sub eax,20h
imul eax,eax,5
mov ecx,9
idiv eax,ecx
mov dword ptr [ebp-8],eax
mov eax,dword ptr [ebp-8]
mov esp,ebp
pop ebp
ret
push ebp
mov ebp,esp
mov eax,dword ptr [ebp+8]
add eax,dword ptr [ebp+0Ch]
mov dword ptr [ebp+8],eax
mov eax,dword ptr [ebp+8]
mov esp,ebp
pop ebp
ret
push ebp
mov ebp,esp
push 64h
call 01A10E6h
add esp,4
mov dword ptr [a],eax
push 10h
push 8
call 01A11D1h
add esp,8
mov dword ptr [b],eax
mov esp,ebp
pop ebp
ret
```

Задача – восстановить исходную программу на языке C++ по части дизассемблированного кода.