Настройки

Управление курсом

Настройки моего профиля

Навигация

В начало

- Моя домашняя страница

Страницы сайта

Мой профиль

Текущий курс

ПИ 3. АК

Участники

ЭУМК «Архитектура компьютеров»

Модуль 1. «Базовые принципы архитектуры»

On-line лекции

1. Введение

2. Архитектура 32-битных Intel-совместимых микропр...

3. Синтаксис языка Ассемблера

4. Система команд микропроцессора Intel 80x86

20.10.2020 - Видеозапись лекции по системе команд ...

5. Подпрограммы

Лабораторная работа №1. Принципы фон Неймана

Лабораторная работа №2. Структура программы на язы...

Лабораторная работа №3. Арифметические операции в ...

Лабораторная работа №4. в языке ...

Лабораторная работа №5. Подпрограммы и массивы в языке Ассемблера

Лабораторная работа №5

Подпрограммы и массивы в языке Ассемблера

Задание 1. Внимательно изучите структуру подпрограммы arrayToStr, предназначенной для формирования строки, содержащей все элементы массива. Подготовьте программу, демонстрирующую использование данной подпрограммы.

```
; ------------- Подпрограмма ArrayToStr ---------------------
; void arrayToStr(char* buffer, int* array, int size)
;
; Формирует строковое представление целочисленного массива
;
; Входные параметры:
;       buffer ( [EBP + 8] )  - указатель на строку, в которой будет
;                               формироваться представление массива
;       array  ( [EBP + 12] ) - указатель на массив
;       size   ( [EBP + 16] ) - количество элементов в массиве

.data
        template db "%d ", 0    ; Образец строки для одного целого числ

.code

arrayToStr:
        ; Стандартный пролог функции
        push EBP
        mov EBP, ESP
        ; начало цикла - пока есть необработанные элементы
        cycle:
                cmp dword ptr [ EBP + 16 ], 0
                je endFunction
                ; Формирование текстового представления целого числа
                mov EAX, [ EBP + 12 ]         ; указатель на очередное
                push [ EAX ]                  ; целое число (элемент ма
                                              ; преобразуемое в строку
                push offset template          ; шаблон строки, в которы
                                              ; подставляются значения
                push [ EBP + 8 ]              ; адрес буфера для размещ
                                              ; итоговой строки
                call wsprintf                 ; в EAX записывается числ
                                              ; записанных в буфер
                add ESP, 12                   ; выровняем стек
                ; Подготовка к следующему числу
                add [ EBP + 8 ], EAX          ; рассчитаем адрес строки
                                              ; следующего числа
                add dword ptr [ EBP + 12 ], 4 ; перемещаем указатель на
                                              ; элемент массива
                dec dword ptr [ EBP + 16 ]    ; уменьшаем счетчик
        ; конец цикла
        jmp cycle
        endFunction:
        ; Стандартный эпилог функции
```

```
        pop EBP
; Выйти и выровнять стек
ret 12
```

Краткая теория:

***Функция* wsprintf**

Записывает форматированные данные в строковый буфер. Все аргументы конвертируются в строку и копируются в выходной буфер согласно шаблона, указанного в строке форматирования. Функция дописывает завершающий строку нулевой символ в конец строкового буфера, но не учитывает этот символ, когда возвращает общее количество записанных в выходную строку символов.

Прототип данной функции на языке C++ выглядит так:

```
int wsprintf(
    LPTSTR lpOut,
    LPCTSTR lpFmt,
    ...
);
```

Параметры функции:

lpOut

Выходной обязательный параметр — адрес начала строкового буфера, в который будут скопированы форматированные данные. Максимальный объём буфера 1024 байта.

lpFmt

Входной обязательный параметр — адрес начала строки форматирования, содержащей шаблон, определяющий, каким образом будут форматироваться данные, копируемые в выходную строку. Подробнее смотрите в пояснении.

...

Входные необязательные параметры — данные — переменные различных типов, форматирование которых будет производиться в соответствии с указаными в строке форматирования шаблонами.

Возвращаемое значение:

Количество символов, скопированных функцией в выходной буфер (в этом количестве не учитывается завершающий нулевой символ, добавляемый в коец строки).

Пояснение:

Функция последовательно читает строку форматирования, и если очередной символ это строки не равен символу '%', то этот символ копируется в выходной буфер. Если же функция из строки форматирования читает символ '%', то функция считывает шаблон, имеющий вид:

%тип

Далее функция читает очередной аргумент из списка аргументов-данных, преобразует значение этого аргумента в строковое представление в соответствии с типом, указанным в шаблоне, и полученную строку вместо самого шаблона копирует в выходную строку. Процесс продолжается до тех пор, пока строка форматирования не закончится. Количество шаблонов в строке форматирования не должно быть больше количества аргументов-данных.

Некоторые возможные значения типа, используемого в шаблоне:

тип	описание

d целое знаковое число в десятичной системе счисления

u целое беззнаковое число в десятичной системе счисления

x целое беззнаковое число в шестнадцатиричной системе счисления (цифры от A до F записываются в нижнем регистре)

X целое беззнаковое число в шестнадцатиричной системе счисления (цифры от A до F записываются в верхнем регистре)

c одиночный символ

s строка

Функция использует переменное число параметров, поэтому она не выравнивает стек самостоятельно. Это нужно сделать после вызова функции вручную, удалив из стека все параметры, переданные в функцию через стек.

Задание 2. Составьте программу, которая обрабатывает целочисленный массив из нескольких (например, 10) элементов следующим образом:

- заполняет массив некоторыми числами (согласно пункту ***a*** соответствующего варианта задания);
- выводит сформированный массив на экран;
- подсчитывает и выводит на экран сумму элементов массива;
- изменяет элементы массива по некоторому правилу (согласно пункту ***b*** варианта задания);
- выводит полученный массив на экран;
- подсчитывает и выводит на экран новую сумму элементов массива.

Заполнение, изменение и подсчет суммы элементов массива необходимо оформить в виде отдельных подпрограмм.

Варианты задания:

1. a. арифметическая прогрессия: $a_0 = 18$, $d = 43$
   b. элементы, кратные четырём, уменьшить в четыре раза
2. a. геометрическая прогрессия: $a_0 = 3$, $q = -3$
   b. каждый отрицательный элемент уменьшить в 3 раза
3. a. числа Фиббоначи: $a_0 = a_1 = 1$, $a_2 = a_1 + a_0$, $a_3 = a_2 + a_1$, ...
   b. чётные элементы возвести в квадрат
4. a. последовательность квадратов натуральных чисел: 1, 4, 9, ...
   b. поменять знак у нечётных чисел
5. a. последовательность кубов чисел, начиная от -5: -125, -64, -27, ...
   b. чётные числа возвести в квадрат
6. a. последовательность чисел, кратных 7: 7, 14, 21, ...
   b. чётные числа уменьшить в 2 раза
7. a. последовательность остатков от деления числа 101 на 1, 2, 3, ...
   b. поменять знак у чётных чисел
8. a. последовательность квадратов чисел, начиная от -10: 100, 81, ...
   b. элементы, заканчивающиеся на 6, увеличить в 3 раза
9. a. последовательность степеней тройки: 3, 9, 27, ...
   b. элементы, заканчивающиеся на 9, увеличить на 1
10. a. факториалы чисел от 1 до N
    b. от каждого элемента массива отнять среднее арифметическое всех элементов массива

Состояние ответа

Состояние ответа на задание	Ответ на задание должен быть представлен вне сайта
Состояние	Не оценено

оценивания	

Вы зашли под именем Никита Иванов (Выход)

ПИ 3. АК