**Лабораторные работы**

Правила выполнения Лабораторной работы

получение навыков работы в среде MS DOS, с компилятором TASM и отладчиком Turbo Debugger.

Каждая лабораторная работа состоит из двух частей. В первой части л.р. приводится пример программы. В программе используются конструкции и операторы, соответствующие теме лабораторной работы. Не нужно присылать описание выполнения этой части л.р. Вторая часть состоит из вариантов заданий для самостоятельного выполнения. Номер задания студент выбирает по последней цифре пароля.

Для зачета по лабораторной работе студенту необходимо представить

* Исходные тексты программ с подробными комментариями;
* Исполняемые файлы;
* Отчет по лабораторной работе (пример отчета приведен ниже).

Отчет обязательно должен включать в себя следующие разделы

* Формулировку задания (текст задачи из второй части л.р.)
* Описание созданной программы;
* Результаты работы программы (в виде файла или в виде скриншота);
* Анализ результатов.

Лабораторная работа будет оценена как незачтеная, если

* Программа не соответствует поставленной задаче или работает неправильно
* Отсутствует отчет, либо в отчете отсутствует описание программы, либо в отчете отсутствуют результаты и выводы
* Отсутствуют исходные и/или исполняемые файлы
* Отсутствуют полностью или частично комментарии в программе

Лабораторная работа 1

Тема: Вычисление арифметических выражений

**Цель работы:** Научиться использовать арифметические команды языка ассемблера.

Задание 1

1. Изучите приведенную ниже программу на языке ассемблера. Программа вычисляет выражение 5\*(3+8\*9) и результат заносит в регистр АХ.

|  |  |
| --- | --- |
| **TITLE** **Program5** | ;название программы |
| **.MODEL small** | ;отводим под стек и под данные по 64Кб |
| **.STACK 100h** | ;отмечаем начало сегмента стека |
| **.DATA** | ;отмечаем начало сегмента данных |
| **A DW 5 ;**  **B DW 3 ;**  **C DW 8 ;**  **D DW 9 ;** |  |
| **.CODE** | ;отмечаем начало сегмента кодов |
| **main PROC** |  |
| **MOV AX, @data** | ;копируем адрес |
| **MOV DS, AX** | ;сегмента данных |
| **mov AX, C**  **mov BX, D**  **mul BX** | ;Умножение 8\*9. Результат в АХ |
| **add AX, B** | ; прибавление 3 В АХ 3+8\*9 |
| **mov BX, A**  **mul BX** | ;умножение на 5 |
| **MOV AX,4C00h** | ;выход |
| **INT 21h** | ;из программы |
| **main ENDP** |  |
| **END main** |  |

1. Выполните отладку программу с использованием отладчика TurboDebugger. В ходе отладки посмотрите как изменяются регистры в процессе выполнения программы. Чему равно значение регистра АХ перед выходом из программы? Совпадает ли результат вычислений с реальным значением выражения?

Задание 2

Используя предложенную выше программу, разработайте программу, которая вычисляет заданное выражение. Исполнить программу с несколькими (три - четыре) наборами исходных данных, проверить правильность результатов. Числа в наборе данных должны быть различными, т.е. положительными, отрицательными, нулевыми. (В отчете обязательно привести результаты вычислений).

Номер задания выбирается в соответствии с последней цифрой пароля.

1. А – 5·(В – 2·С) + 2

Замечание. Переменные A, B, C являются целочисленными, могут быть положительными либо отрицательными. Деление полагается целочисленным, т.е. например, 5/2=2, 6/2=3.

### Лабораторная работа 2

### Тема: Оператор цикла LOOP. Команды условных переходов

**Задание 1**

1. Изучите приведенную ниже программу на языке ассемблера. Программа суммирует числа от 1 до 10 и результат заносит в регистр АХ.

|  |  |
| --- | --- |
| **TITLE** **Program2** | ;название программы |
| **.MODEL small** | ;отводим под стек и под данные по 64Кб |
| **.STACK 100h** | ;отмечаем начало сегмента стека |
| **.DATA** | ;отмечаем начало сегмента данных |
| **.CODE** | ;отмечаем начало сегмента кодов |
| **main PROC** |  |
| **MOV AX, @data** | ;копируем адрес |
| **MOV DS, AX** | ;сегмента данных |
| **CLD** |  |
| **xor AX,AX**  **mov CX, 10**  **L1:**  **add AX, CX**  **loop L1** | ; здесь в цикле суммируются числа  ; начиная с 10 и заканчивая 1 |
| **MOV AX,4C00h** | ;выход |
| **INT 21h** | ;из программы |
| **main ENDP** |  |
| **END main** |  |

1. Выполните отладку программу с использованием отладчика TurboDebugger. Посмотрите как изменяются регистры в процессе выполнения программы.Чему равно значение регистра АХ перед выходом из программы? регистра СХ? Совпадает ли результат вычислений с реальным значением выражения?

Задание 2

Используя предложенную выше программу, выполните второе задание. Исполнить программу с несколькими (три - четыре) значениями параметра N, проверить правильность результатов. В отчете обязательно привести результаты вычислений.

* 1. Составьте программу для подсчета суммы 1+3+5+…+(2·N+1), где N – произвольное положительное число. В программе нельзя использовать команды условных переходов. Допускается использование оператора цикла.

### Лабораторная работа 3

### Тема: Работа с массивами данных

Цель работы: Научиться обрабатывать массивы данных.

**Задание 1**

1. Изучите приведенную ниже программу на языке ассемблера. Программа вычисляет максимальный элемент массива и результат заносит в переменную регистр Max.

|  |  |
| --- | --- |
| **TITLE Program3** | ;название программы |
| **.MODEL small** | ;отводим под стек и под данные по 64Кб |
| **.STACK 100h** | ;отмечаем начало сегмента стека |
| **.DATA** | ;отмечаем начало сегмента данных |
| **A DW 5, 3, 9, 4, 5**  **Max DW 0** | ;описание массива из 5 элементов  ;описание переменной **Max** |
| **.CODE** | ;отмечаем начало сегмента кодов |
| **main PROC** |  |
| **mov AX, @data** | ;копируем адрес |
| **mov DS, AX** | ;сегмента данных |
| **mov SI, OFFSET A** | ;заносим в SI начало массива А |
| **mov CX, 4** | ;в СХ заносим количество итераций |
| **cld** | ;устанавливаем прямой порядок обработки массива |
| **mov AX, [SI]** |  |
| **mov Max, AX**  **L1:** | ;в Мах заносим первый элемент массива |
| **inc SI** | ;сдвигаемся по массиву к следующему |
| **inc SI** | ;элементу |
| **mov AX,[SI]** | ;заносим в АХ текущий элемент массива |
| **cmp AX, Max**  **jbe L2** | ;сравниваем АХ с Мах  ;если меньше или равно, переходим на **L2** |
| **mov Max, AX**  **L2:** | ;иначе запоминаем новый максимум |
| **loop L1** | ;циклически повторяем все действия |
| **mov AX,4C00h** | ;выход |
| **int 21h** | ;из программы |
| **main ENDP** |  |
| **END main** |  |

1. Выполните отладку программу с использованием отладчика TurboDebugger. Посмотрите как изменяются регистры в процессе выполнения программы.Чему равно значение переменной Max перед выходом из программы? Совпадает ли результат вычислений с реальным значением максимального элемента массива?

Задание 2

Используя предложенную выше программу, выполните второе задание. Исполнить программу с несколькими (три - четыре) различными (по количеству и по составу) массивами данных, проверить правильность результатов.

1. Составьте программу, которая вычисляет и заносит в DX номер максимального элемента массива.

**Контрольной работы**

**При выполнении контрольной работы** необходимо строго придерживаться указанных ниже правил. Работы, выполненные без соблюдения этих правил, не засчитываются и возвращаются студенту для переработки.

При выполнении работы следует использовать операционную среду MS-DOS (или режимы эмуляции MS-DOS) и компилятор языка ассемблера TASM (или MASM).

Перед решением каждой задачи необходимо выписать полностью ее условие. В том случае, если несколько задач, из которых студент выбирает задачу своего варианта, имеют общую формулировку, следует, при переписывании условия задачи, заменить общие данные конкретными из соответствующего номера.

В контрольную работу необходимо включить распечатки командного файла и файла с программой на языке ассемблера (файлы должны быть снабжены комментариями), а также результаты вычислений. Кроме этого в работу должны быть включены исполняемый bat-файл (1 задание) и exe-файл (2 задание) Также см. правила выполнения и оформления л.р.

Задание:

**Написать и отладить программу на языке ассемблера. В программе описать процедуру, которая вычисляет число четных элементов в массиве. Параметры передавать следующим образом:**

в ВХ – смещение массива;

в СХ – число элементов в массиве;

в АХ – результат вычислений (число четных элементов).

В основной программе вызвать описанную процедуру для двух разных массивов.