МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт (факультет) | Информационных технологий |
| Кафедра | Математическое и программное обеспечение ЭВМ |

КУРСОВАЯ РАБОТА

|  |
| --- |
| по дисциплине Программирование на ассемблере |

|  |  |
| --- | --- |
| на тему | «Программирование на языке низкого уровня» |
|  | |

|  |
| --- |
| Выполнил студент группы |
| 1ПИб-02-1оп-22 |
| направления подготовки (специальности) |
| 09.03.04, Программная инженерия |
| шифр, наименование |
|  |
| Тенигина Виктория Романовна |
| фамилия, имя, отчество |

|  |
| --- |
| Руководитель |
| Виноградова Людмила Николаевна |
| фамилия, имя, отчество |
| Доцент |
| должность |

|  |
| --- |
| Дата представления работы |
| «\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 г. |
|  |
| Заключение о допуске к защите |
|  |
|  |
|  |
|  |
| Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| количество баллов |
| Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Череповец, 2024

Год

Аннотация

Данная курсовая работа посвящена программированию на языке низкого уровня Assembler. В работе рассматривается написание программы, которая подсчитывает количество структур, подходящих по заданному условию.

В данной работе описывается алгоритм созданной программы, а также содержится описание процессора 8086 и основных команд, использованных в процессе разработки.

Оглавление

[Введение 4](#_Toc182915886)

[Описание предметной области 5](#_Toc182915887)

[Логическое проектирование 8](#_Toc182915888)

[Физическое проектирование 9](#_Toc182915889)

[Кодирование 10](#_Toc182915890)

[Тестирование 12](#_Toc182915891)

[Заключение 14](#_Toc182915893)

[Список литературы 15](#_Toc182915894)

[Приложение 1. 16](#_Toc182915895)

[Приложение 2. 21](#_Toc182915896)

[Приложение 3. 25](#_Toc182915897)

Введение

В современном мире существует немало языков программирования. Одним из самых первых является Assembler. В 1972 году Intel разработала процессор 8008, и в 1978 году выпустила процессор 8086, который стал одним из самых важных моментов в истории процессоров x86. Процессор 8086 был разработан для использования в персональных компьютерах IBM PC и стал первым процессором семейства x86, который расширил возможности программирования с использованием языка ассемблера.

Язык ассемблера для процессора 8086 стал главным инструментом разработки программ для IBM PC. Сочетая в себе гибкость и прямую работу с аппаратными ресурсами, язык ассемблера позволил создавать более эффективные и быстрые программы для компьютеров.

С течением времени, процессоры 8086 и их последователи стали основой для развития компьютерной технологии, и язык ассемблера продолжал быть важным инструментом при разработке низкоуровневых системных программ. С развитием технологий и появлением более мощных процессоров, язык ассемблера продолжает играть важную роль в разработке программного обеспечения, особенно в области встроенных систем и высокопроизводительных вычислений.

Описание предметной области

Предметной областью является набор команд для ассемблера x86, в основном команды для работы с массивами и структурами

Команды пересылки данных выполняют операции обмена данными между регистрами и ячейками памяти. Команда MOV пересылает байт или слово из ячеек памяти или регистров. Оба операнда должны иметь одинаковый размер, например, байт или слово.

Общий вид команды:

MOV <приемник>, <источник>

Источник занимает место приемника, при этом приемник теряется. Приемником может быть регистр общего назначения, сегментный регистр (исключая регистр CS) или ячейки памяти. Источником также может быть и константой.

Структура - это тип данных, который позволяет объединить различные типы данных в один составной объект. В ассемблере структуры могут быть реализованы различными способами, в зависимости от используемой архитектуры и синтаксиса языка.

Общий вид типа структуры:

<имя типа> STRUC

<имя поля>

…………..

<имя типа> ENDC

Структуры в ассемблере могут быть определены с помощью директив, таких как db, dw, dd и т.д. Данные директивы позволяют задать структуру данных, определить ее поля и их типы.

Для записи множества элементов одинакового типа используются массивы. Массив в языке ассемблер представляет собой набор последовательных ячеек памяти, каждая из которых содержит определенное значение. Каждая ячейка массива имеет свой уникальный адрес, по которому осуществляется доступ к данным.

Для создания массива в ассемблере выделяется блок памяти достаточного размера для хранения всех элементов. После этого каждый элемент массива инициализируется соответствующими значениями. Для доступа к отдельным элементам массива используются индексы или указатели на ячейки памяти.

Главный способ инициализации – это перечислением элементов массива в поле операндов одной из директив описания данных. При перечислении элементы разделяются запятыми. Например, массив из 10 элементов, размер каждого элемента 2 байта: M1 dw 1,2,3,4,5,6,7,8,9,10

Команды условных переходов позволяют передать управление процессора на указанную команду в зависимости от выполнения некоторых условий, которые, в свою очередь, определяются состоянием флагов. Команда безусловного перехода передает управление всегда (не имеет условий). Каждая команда условного или безусловного перехода имеет один операнд, определяющий команду, на которую должно передаваться управление. В качестве такого операнда выступает адресное выражение.

Синтаксис команд условного и безусловного переходов:

<команда\_перехода> <адресное\_выражение>

Для формирования флагов совместно с командами условных переходов используется команда CMP. Команда CMP сравнивает два операнда, вычитая второй операнд из первого, и устанавливает флаги в зависимости от результатов сравнения. Команда CMP, как правило, ставится сразу перед командой условного перехода.

В таблице 1 приведен список всех команд условных и безусловных переходов, а также условия (флаги), которые эти команды проверяют.

Из представленных ниже команд были использованы команды безусловного перехода (JMP), больше (JB).

Таблица 1

Команды условных и безусловных переходов

|  |  |  |
| --- | --- | --- |
| **Мнемоническое обозначение** | **Условие перехода** | **Проверяемые  значения флагов** |
| JMP | безусловный переход | Нет |
| JA / JNBE | больше (для беззнаковых чисел) | CF=0 и ZF=0 |
| JAE / JNB | больше или равно (для беззнаковых чисел) | CF=0 |
| JB / JNAE | меньше (для беззнаковых чисел) | CF=1 |
| JBE / JNA | меньше или равно (для беззнаковых чисел) | CF=1 или AF=1 |
| JC | перенос | CF=1 |
| JCXZ | CX равно нулю | нет |
| JE / JZ | равно | ZF=1 |
| JG / JNLE | больше (для знаковых чисел) | ZF=0 и SF и OF одинаковы (оба 0 или 1) |
| JGE / JNL | больше или равно (для знаковых чисел) | SF и OF одинаковы (оба 0 или 1) |
| JL / JNGE | меньше (для знаковых чисел) | SF и OF различны |
| JLE / JNG | меньше или равно (для знаковых чисел) | ZF=1 и SF и OF различны |
| JO | переполнение | OF=1 |
| JNO | нет переполнения | OF=0 |
| JP / JPE | паритет четный | PF=1 |
| JNP / JPO | паритет нечетный | PF=0 |
| JS | знак | SF=1 |
| JNS | нет знака | SF=0 |
| JNZ | не нуль | ZF=0 |

Логическое проектирование

Дана структура Cars с полями: фирма-изготовитель, модель, объем двигателя, цена и массив S типа структуры, где собрана информация о 10 машинах. Записать в регистр BP количество машин с объемом двигателя 1.6 л.

Вначале создаем шаблон структур с параметрами, описанными в задании. После чего инициализируем массив из 10 заполненных структур. Создаём переменную, в которой будет подсчитываться количество машин и переменную для подсчета структур подходящих по условию. Определяется переменная для перехода между элементами массива. Данная переменная будет, увеличивается на размер 1 структуры для перехода между элементами массива.

Далее проверяем условие. После чего переходим к следующему элементу массива, увеличивая переменную подсчитывающую количество машин. В конце записываем полученный результат в отдельную переменную.

Описание алгоритма работы:

1. Создание шаблона структуры.
2. Инициализация массива и заполнение его 10 структурами.
3. Пока количество обработанных машин не равно 10:
   1. Сравнить объем двигателя с условием (объем двигателя = 1.6).
   2. В случае прохождения условия увеличить количество подходящих машин на 1.
   3. Увеличение количества обработанных структур на 1.
   4. Переход к следующему элементу массива.
4. Запись окончательного ответа.
5. Завершение работы программы.

Физическое проектирование

Шаблон структуры, используемой в программе:

Cars struc

LN db 15 dup (' ')

T db 20 dup (' ')

Y dw ?

P dw ?

Cars ENDS

Где Cars – название структуры, LN – фирма-изготовитель машины, T – модель машины, Y – объем двигателя \* 10, P – цена машины, указанная в тысячах долларов.

Инициализируем массив B представленный таким образом: S Cars <…..>, … ,<…..>. Где S – название массива, Cars – тип данных использованный в массиве, а именно структура, инициализированная ранее, в <…..> записываются все элементы структуры через “,”.

Регистр BX, используется как счетчик подходящих по условию структур. Регистр увеличивается с помощью команды INC.

Регистр CX, используется как счетчик уже обработанных структур. Регистр увеличивается с помощью команды INC.

Регистр SI, используется для обращения к следующему элементу массива. Чтобы это происходило, мы увеличиваем регистр на размер структуры в байтах с помощью команды ADD.

Регистр SP, используется для записи в него конечного результат работы программы.

Метка SECOND – В данной метке производится взятие элемента Y из новой структуры, а также сравнение с условием.

Метка PLUS – В данной метке увеличивается регистр BX.

Метка EN – В данной метке сравнивается количество уже обработанных структур.

Кодирование

В данном разделе представлены последовательности команд, использованных для нахождения количества машин, которые имеют объем двигателя равный 1,6 л.

Инициализация шаблона структуры Cars:

Cars struc

LN db 15 dup (' ')

T db 20 dup (' ')

Y dw ?

P dw ?

Cars ENDS

Инициализация массива структур в Data segment:

DATA\_SEG SEGMENT

S Cars <'Volkswagen','Jetta', 10, 6>, <'Porsche','Cayenne', 44, 70>, <'Lada','Vesta', 16, 18>, <'Chevrolet','Cruze', 18, 8>, <'BMW','i3', 8, 25>, <'Kia','Rio', 16, 12>, <'Land Rover','Range Rover Sport', 30, 60>, <'Lexus','RX', 44, 90>, <'Renault','Duster', 13, 22>, <'Peugeot','308', 18, 5>

DATA\_SEG ENDS

Подключаем Data segment к основному коду:

ASSUME CS: CODE\_SEG, DS:DATA\_SEG

START:

MOV AX, DATA\_SEG

MOV DS, AX

Очистка регистров для дальнейшей работы с ними:

XOR CX,CX

XOR SI,SI

XOR BX,BX

В данной части мы берем новый элемент массива и обрабатываем его, увеличивая количество обработанных слов. С помощью условий перехода определяем, подходит элемент или нет:

SECOND:

INC CX

MOV AX, S[SI].Y

ADD SI, 39

CMP AX, 16

JЕ PLUS

JMP EN

В данной части инкрементируем регистр BX для подсчета подходящих по условию структур:

PLUS:

INC BX

JMP EN

В данной части проверяем остались ли структуры для обработки, если нет, то записываем в регистр BP результат и заканчиваем работу программы:

EN:

CMP CX, 10

JB SECOND

MOV BP,BX

MOV AH, 4Ch

INT 21h

Тестирование

В таблице 2 приведены тестовые данные, а в таблице 3 результаты тестов.

Таблица 2

Тестовые данные

|  |  |  |
| --- | --- | --- |
| Исходные данные | Тестируемый модуль | Ожидаемые результаты |
| <'Volkswagen','Jetta', 10, 6>,  <'Porsche','Cayenne', 44, 70>,  <'Lada','Vesta', 16, 18>,  <'Chevrolet','Cruze', 18, 8>,  <'BMW','i3', 8, 25>,  <'Kia','Rio', 16, 12>,  <'Land Rover','Range Rover Sport', 30, 60>,  <'Lexus','RX', 44, 90>,  <'Renault','Duster', 13, 22>,  <'Peugeot','308', 18, 5> | CURS.asm | 2 |
| <'Volkswagen','Jetta', 10, 6>,  <'Porsche','Cayenne', 44, 70>,  <'Lada','Vesta', 16, 18>,  <'Chevrolet','Cruze', 16, 8>,  <'BMW','i3', 8, 25>,  <'Kia','Rio', 16, 12>,  <'Land Rover','Range Rover Sport', 30, 60>,  <'Lexus','RX', 44, 90>,  <'Renault','Duster', 13, 22>,  <'Peugeot','308', 16, 5> | CURS.asm | 4 |

Таблица 3

Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Дата тестирования | Тестируемый модуль | Кто проводил тестирование | Описание теста | Результаты тестирования |
| 20.10.2024 | CURS.asm | Тенигина В.Р. | Массив S с информацией о 10 машинах (<'Volkswagen','Jetta', 10, 6>, <'Porsche','Cayenne', 44, 70>, <'Lada','Vesta', 16, 18>, <'Chevrolet','Cruze', 18, 8>, <'BMW','i3', 8, 25>, <'Kia','Rio', 16, 12>, <'Land Rover','Range Rover Sport', 30, 60>, <'Lexus','RX', 44, 90>, <'Renault','Duster', 13, 22>, <'Peugeot','308', 18, 5>), и его обработка с помощью программы. | 2 |
| 21.10.2024 | CURS.asm | Тенигина В.Р. | Массив S с информацией о 10 машинах (<'Volkswagen','Jetta', 10, 6>, <'Porsche','Cayenne', 44, 70>, <'Lada','Vesta', 16, 18>, <'Chevrolet','Cruze', 16, 8>, <'BMW','i3', 8, 25>, <'Kia','Rio', 16, 12>, <'Land Rover','Range Rover Sport', 30, 60>, <'Lexus','RX', 44, 90>, <'Renault','Duster', 13, 22>, <'Peugeot','308', 16, 5>), и его обработка с помощью программы. | 4 |

Заключение

В результате выполнения данной курсовой работы была создана программа на языке Assembler, предназначенная для обработки массива структур. Программа подсчитывает количество машин, описанных с помощью структур, которые имеют объем двигателя равный 1.6 л.

Таким образом, разработанная программа соответствует целям и задачам и обеспечивает эффективную обработку массивов структур.

В ходе разработки было получено ещё более глубокое понимание архитектуры компьютера, а также сред разработки на языке ассемблер.

Список литературы

1. Е.В. Ершов, Л.Н. Виноградова, В.В. Селивановских, О.Л. Селяничев «Методика и организация самостоятельной работы студентов»;
2. Л.Н. Виноградова, учебное пособие «Системное программирование»;
3. История языка ассемблер и процессора 8086 [Электронный ресурс].   
   URL: <https://habr.com/ru/articles/417983/>   
   (дата обращения:26.12.2023)
4. Индексация массивов, структуры в ассемблере [Электронный ресурс].   
   URL: <https://asmforfun.blogspot.com/2009/06/111.html>   
   (дата обращения:26.12.2023)

# Приложение 1.

МИНОБОРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

наименование института (факультета)

Математическое и программное обеспечение ЭВМ

наименование кафедры

Программирование на ассемблере

наименование дисциплины в соответствии с учебным планом

УТВЕРЖДАЮ

Зав. кафедрой МПО ЭВМ

д.т.н., профессор Ершов Е.В.

« » 2023 г.

Программирование на языке низкого уровня

Техническое задание на курсовую работу

Листов 9

Руководитель: Виноградова Л.Н

Исполнитель: студент гр. 1ПИб-02-1оп-22

Тенигина В.Р.

2024 год

Введение

Данная программа нацелена на реализацию вычисления количества объектов, с заданным параметром основываясь на переданных в подпрограмму данных.

1. Основания для разработки

Основанием для разработки является задание на курсовую работу по дисциплине "Программирование на ассемблере", выданное на кафедре МПО ЭВМ ИИТ ЧГУ. Наименование темы разработки: "Программирование на языке низкого уровня"

2. Назначение разработки

Основная цель исследования курсовой работы является реализация программы, которая будет работать структурами и определять кол-во параметров с заданной величиной.

1. Требования к программе

Программа должна соответствовать требованиям, изложенным в П.1.3.1 – П.1.3.8 разделах.

3.1. Требования к функциональным характеристикам

Программа должна соответствовать следующим характеристикам:

1. Содержать структуру Cars с полями: фирма-изготовитель, модель, объем двигателя, цена.
2. Массив из 10 структур, описанных в предыдущем требовании.
3. Отображение результата работы программы в регистр BP.
4. Подсчёт количества машин, подходящих по условию “с объемом двигателя 1.6 л”.

3.2. Требования к надежности

В ходе проектирования программы нужно учитыватьколичество структур и способ их обработки для правильной работы программы**.**

3.3. Условия эксплуатации

* Операционная система: Программа должна быть разработана и оптимизирована для запуска в операционных системах, совместимых с архитектурой платформы, поддерживающей ассемблер.
* Ввод и вывод: Программа должна иметь поддержку ввода и вывода данных с использованием стандартных устройств.
* Масштабируемость: Программа должна быть способна быстро и эффективно работать с различными объемами входных данных.

3.4. Требования к составу и параметрам технических средств

Компьютер должен удовлетворять минимальным системным требованиям:

1. Процессор с частотой не менее 1 ГГц.
2. Разрядность системы 32 бит или 64 бит.
3. ОЗУ объемом не менее 256 Мб.
4. Жесткий диск со свободным местом.
5. Наличие устройств вывода в виде монитора.
6. Наличие устройств ввода в виде клавиатуры и компьютерной мыши.

3.5. Требования к информационной и программной совместимости

Для использования программы необходимо установить эмулятор GUI Turbo Assembler, а также иметь операционную систему: Windows 2000 / 98 / XP 7 / 8 / 10 / 11 и выше.

3.6. Требования к маркировке и упаковке

Программа должна быть удобно скомпонована, чтобы её можно было легко распаковать и запустить на компьютере пользователя.

3.7. Требования к транспортированию и хранению

Все файлы, отвечающие за работу программы, нужно записать на отдельный информационный накопитель для возможности быстрого восстановления работоспособности программы в случае потери главных файлов.

3.8. Специальные требования

Программа должна соответствовать требованиям ГОСТ Р ИСО 9127-94

4. Требование к программной документации

Программная документация должна содержать расчётно-пояснительную записку с содержанием:

* Титульный лист
* Оглавление
* Введение
* Описание предметной области
* Проектирование программы
* Создание программы
* Заключение
* Список литературы
* Техническое задание
* Руководство пользователя
* Текст программы

4.3. Требования к оформлению

Оформление документации пишется по правилам, перечисленным ниже (табл. П.1.1).

Таблица П.1.1

Правила оформления документации

|  |  |
| --- | --- |
| Документ | Печать на отдельных листах формата А4 (210х297 мм); оборотная сторона не заполняется; листы нумеруются. Печать возможна ч/б.  Файлы предъявляются на компакт-диске: РПЗ с ТЗ; программный код.  Листы и диск в конверте вложены в пластиковую папку скоросшивателя. |
| Страницы | Ориентация – книжная; отдельные страницы, при необходимости, альбомная.  Поля: верхнее, нижнее – по 2 см, левое – 3 см , правое – 1 см. |
| Абзацы | Межстрочный интервал – 1.5, перед и после абзаца – 0. |
| Шрифты | Кегль – 14. В таблицах шрифт 12. Шрифт листинга – 10 (возможно в 2 колонки). |
| Рисунки | Подписывается под ним по центру: Рис.Х. Название  В приложениях: Рис.П1.3. |
| Таблицы | Подписывается: над таблицей, выравнивание по правому: «Таблица Х».  В следующей строке по центру Название  Надписи в «шапке» (имена столбцов, полей) – по центру.  В теле таблицы (записи) текстовые значения – выровнены по левому краю, числа, даты – по правому. |

5.Стадии и этапы разработки программы.

Стадии и этапы разработки представлены в таблице П.1.2.

Таблица П.1.2

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование  этапа разработки | Сроки разработки | Результат выполнения | Отметка о выполнении |
| Определение темы курсовой работы | 08.11.23 | Утверждена тема для разработки |  |
| Оформление технического задания | 17.11.23 | Выполненное технического задания |  |
| Разработка алгоритма | 19.9.24 | Готовый алгоритм |  |
| Написание программы | 10.10.24 | Готовая программа |  |
| Тестирование программы | 11.10.24 | Отлаженная программа |  |
| Оформление РПЗ | 11.11.24 | Написание РПЗ |  |

6. Порядок контроля и приемки

В таблице П.1.3 представлены порядок контроля и приемки.

Таблица П.1.3

Порядок контроля и приемки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование  контрольного этапа  выполнения  курсовой работы | Сроки  контроля | Результат выполнения | Отметка о приемке  результата  контрольного этапа |
| Техническое задание | 17.11.23 | Оформленное техническое задание |  |
| Расчётно-пояснительная записка | 11.11.24 | Оформленная РПЗ |  |
| Защита курсовой работы | 19.11.24 | Итоговая оценка за курсовую работу |  |

# Приложение 2.

1. Общие сведения о программе

Это компьютерная программа, подсчитывает количество машин с объемом двигателя 1.6 л. Вся информация о машинах находится в массиве из структур, в которых содержится информация о машине.

1. Описание установки

Для работы программы нужно установить GUI Turbo Assembler (рис. П.2.1) и доступ к файлу.

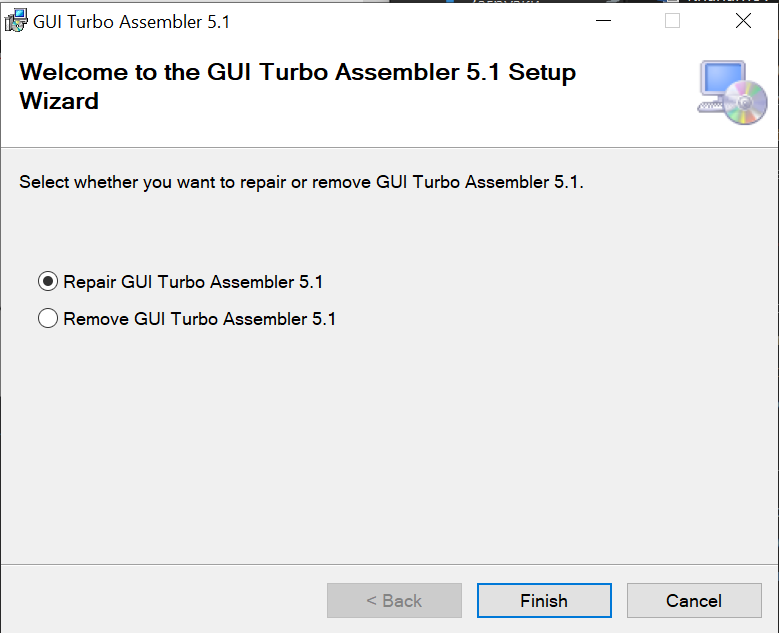


Рис. П.1.1 Окно установки компонентов

3.Описание запуска

Вставьте диск в компьютер. Для запуска программы необходимо открыть файл “CURS.asm”, используя эмулятор (рис. П.2.2).

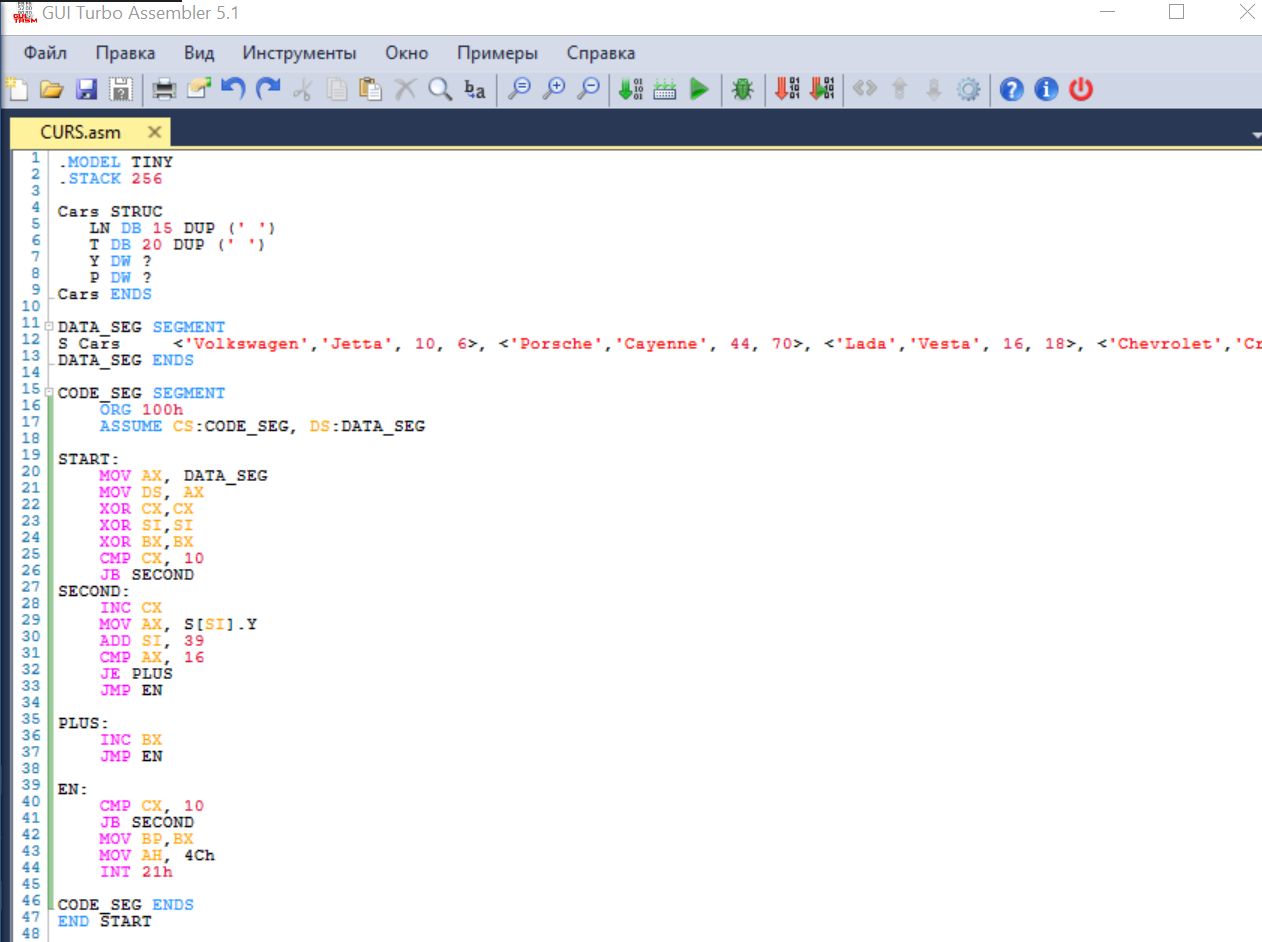


Рис. П.2.2 Программа, запущенная в эмуляторе

Вначале, надо ввести набор данных о машинах в Data segment (рис. П.2.3), для последующей обработки. Ввод данных о машине осуществляется в формате: <’Название марки’, ‘Название модели’, Объём двигателя / 10, л, Цена тыс. долл.>

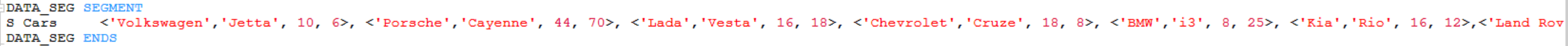


Рис. П.2.3 Поле для изменения информации о машинах

Далее в верхней панели открыть вкладку “инструменты” и выбрать метод “отладка” (рис. П.2.4).

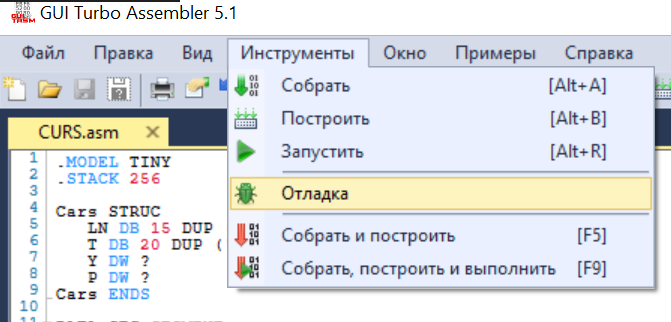


Рис. П.2.4 Вкладка “инструменты” и метод “отладка”

После чего в появившемся окне открыть вкладку с показателями регистров (рис. П.2.5).

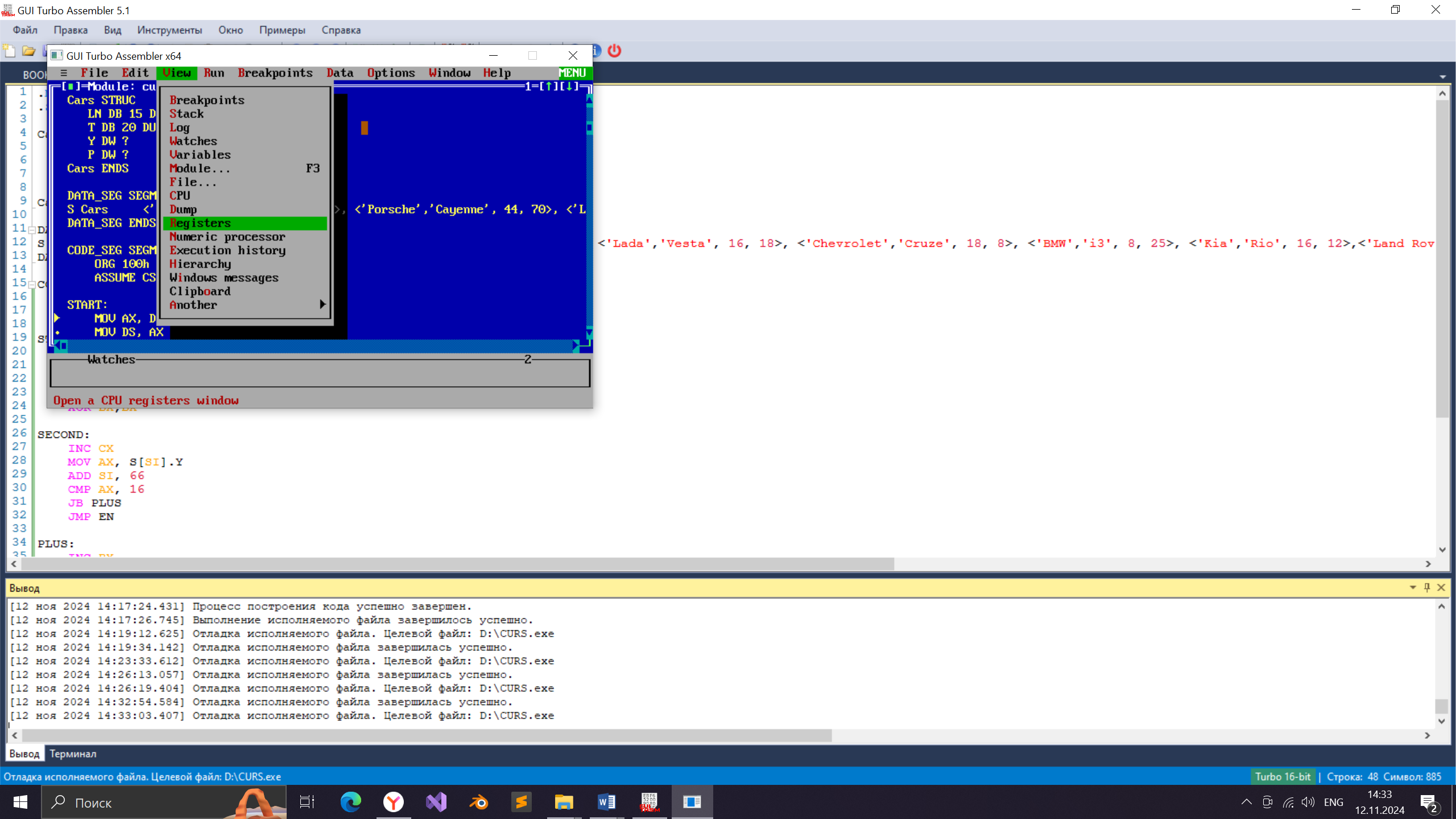


Рис. П.2.5 Открытие вкладки со значения регистров

При нажатии кнопки F8 выполняется 1 шаг программы. Нажатие кнопки F9 выполняет программу полностью. Результат работы виден в окне с регистрами в строчке регистра BP (рис.П.2.6.).

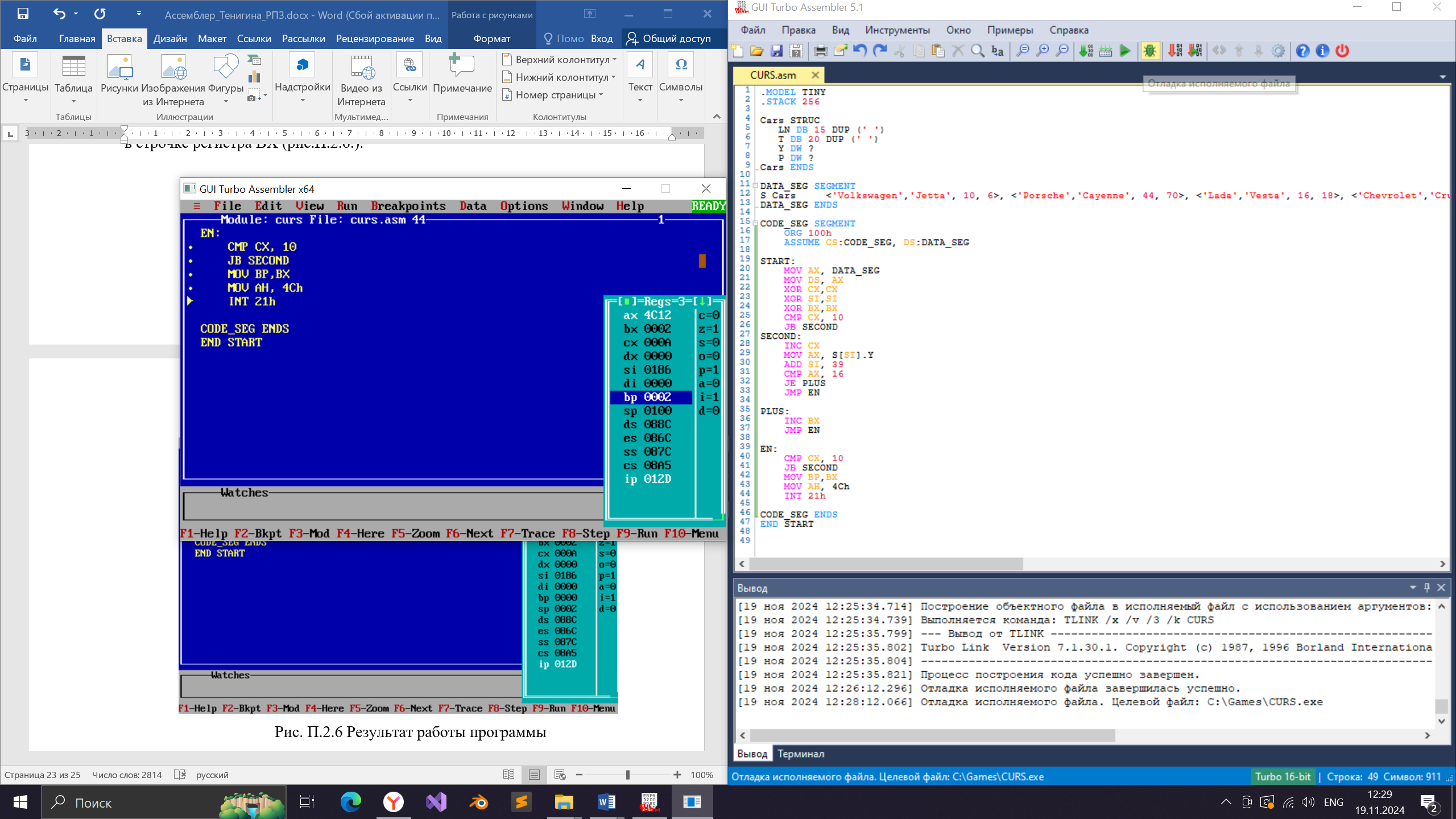


Рис. П.2.6 Результат работы программы

# Приложение 3.

.MODEL TINY

.STACK 256

Cars STRUC

LN DB 15 DUP (' ')

T DB 20 DUP (' ')

Y DW ?

P DW ?

Cars ENDS

DATA\_SEG SEGMENT

S Cars <'Volkswagen','Jetta', 10, 6>, <'Porsche','Cayenne', 44, 70>, <'Lada','Vesta', 16, 18>, <'Chevrolet','Cruze', 18, 8>, <'BMW','i3', 8, 25>, <'Kia','Rio', 16, 12>, <'Land Rover','Range Rover Sport', 30, 60>, <'Lexus','RX', 44, 90>, <'Renault','Duster', 13, 22>, <'Peugeot','308', 18, 5>

DATA\_SEG ENDS

CODE\_SEG SEGMENT

ORG 100h

ASSUME CS:CODE\_SEG, DS:DATA\_SEG

START:

MOV AX, DATA\_SEG

MOV DS, AX

XOR CX,CX

XOR SI,SI

XOR BX,BX

CMP CX, 10

JB SECOND

SECOND:

INC CX

MOV AX, S[SI].Y

ADD SI, 39

CMP AX, 16

JE PLUS

JMP EN

PLUS:

INC BX

JMP EN

EN:

CMP CX, 10

JB SECOND

MOV BP,BX

MOV AH, 4Ch

INT 21h

CODE\_SEG ENDS

END START