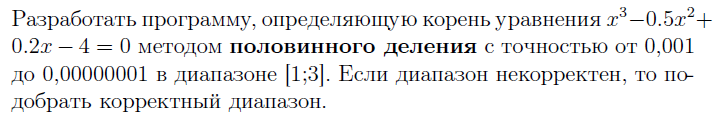
Сёмкин Арсений Сергеевич, БПИ-237

**Архитектура вычислительных систем, ИДЗ-2, вариант 33**

1. **Требования к ДЗ**



Метод половинного деления предполагает, что есть функция и мы знаем такой отрезок, что на его концах функция имеет разные знаки, а внутри него ровно 1 корень. Далее мы за одну итерацию берем середину отрезка и делаем ее новой левой или правой границей так, чтобы на концах нового отрезка функция также имела разные знаки, после чего переходим к следующей итерации. Алгоритм продолжается до тех пор, пока не будет достигнута желаемая точность. ([Источник информации](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%B1%D0%B8%D1%81%D0%B5%D0%BA%D1%86%D0%B8%D0%B8)).

1. **Критерии на 4-5**

• **Приведено решение задачи на ассемблере. Ввод данных осуществ-**

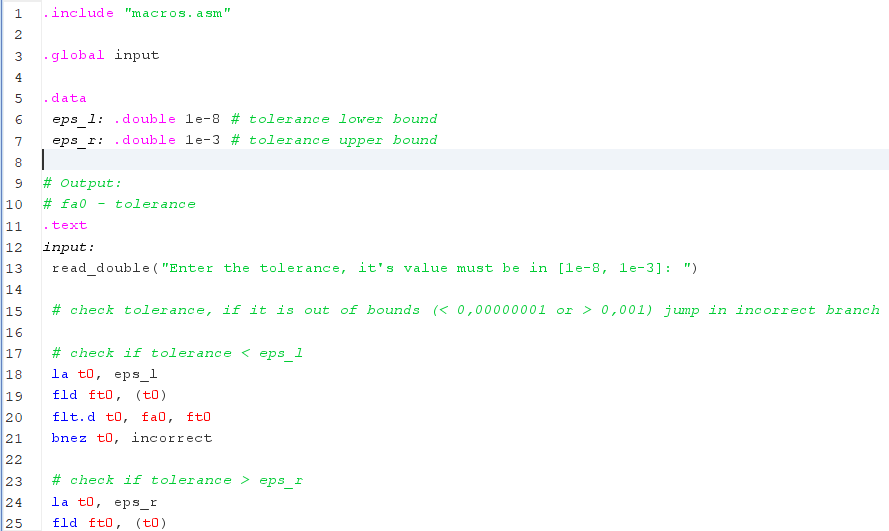
**ляется с клавиатуры. Вывод данных осуществляется на дисплей.**

Выполнено.

• **В программе должны присутствовать комментарии, поясняющие**

**выполняемые действия.**

Их очень много:



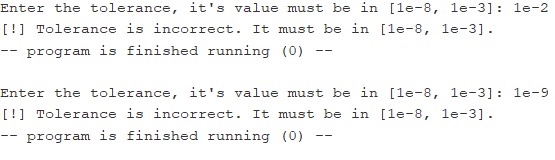
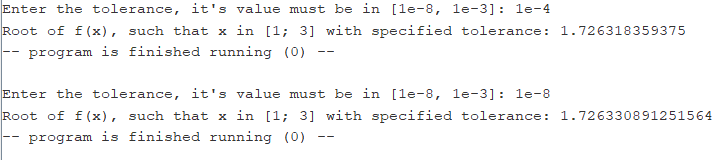
• **Допускается использование требуемых подпрограмм без параметров и локальных переменных.**

Поскольку я выполнял критерии на 10 баллов, подпрограммы получают и возвращают данные согласно конвенции через регистры a. Локальные переменные всегда находятся в t в подпрограммах, могут быть в s в главной программе. Более подробно будет раскрыто далее.

• **В отчете должно быть представлено полное тестовое покрытие.**

**Приведены результаты тестовых прогонов. Например, с использованием скриншотов.**

Были совершены различные тестовые прогоны. Более детально это будет описано в разделе про автоматическое тестирование. Здесь вижу смысл отразить тесты, обрабатывающие входные данные на корректность.

Как видно, случаи с некорректным значением погрешности успешно обрабатываются. При этом в случае корректного значения, ответ успешно вычисляется, даже когда это значение является краевым случаем, то есть лежит на границе допустимого.

1. **Критерии на 6-7**

• **В программе необходимо использовать подпрограммы с передачей**

**аргументов через параметры, что должно обеспечивать их повтор-**

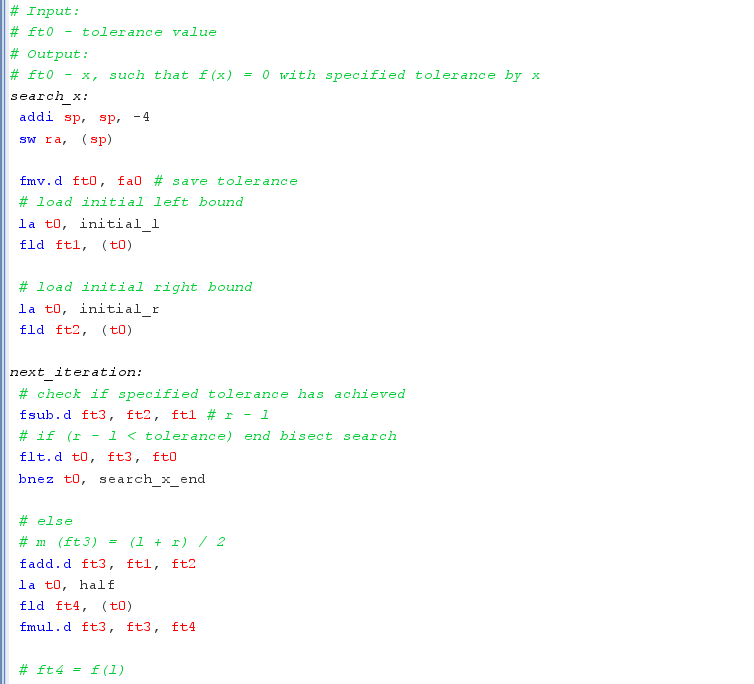
**ное использование с различными входными аргументами, включая,**

**например, применение в других программах. При нехватке реги-**

**стров, используемых для передачи параметров, оставшиеся пара-**

**метры передавать через стек.**

Как уже было описано выше, все регистры используются согласно конвенции. В программе используются подпрограммы, они получают аргументы и возвращают результат через соответствующие регистры a, а для внутренних действий с изменениями задействуют только регистры t. Соответственно эти подпрограммы можно использовать повторно.



Вот, например, отрывок подпрограммы, выполняющей поиск корня функции, половинным делением. Внутри она вызывает подпрограмму для нахождения значения функции с нужным значением аргумента.

**• Внутри подпрограмм необходимо использовать локальные перемен-**

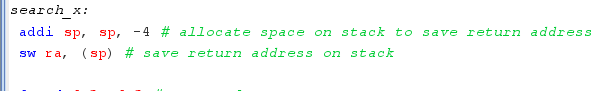
**ные или свободные регистры. То есть, подпрограмма должна быть**

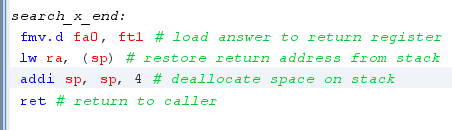
**полностью отделена от вызывающего ее кода. При нехватке вре-**

**менных регистров обеспечить сохранение данных на стеке в соот-**

**ветствии с соглашениями, принятыми для процессора.**

Все локальные переменные в подпрограммах сохраняются в регистрах t. Не пришлось сохранять их на стеке, потому что всегда хватало регистров t. Однако иногда приходилось немало подумать, как использовать эти регистры, чтобы их хватило и не пришлось выделять память для переменных на стеке. Также стоит добавить, что на стеке сохраняются адреса возврата для подпрограмм, которые внутри себя так же вызывают подпрограммы. Когда подпрограмма завершает работу, со стека подгружается адрес возврата, затем он очищается со стека и происходит вовзрат к месту, откуда подпрограмма была вызвана.





**• В местах вызова функции добавить комментарии, описывающие пе-**

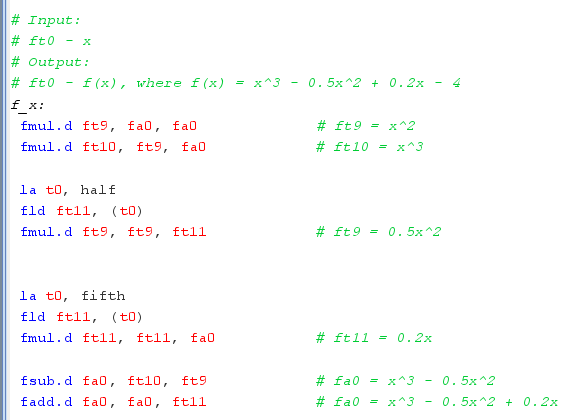
**редачу фактических параметров и местоположение возвращаемого**

**результата. При этом необходимо отметить, какая переменная или**

**результат какого выражения соответствует тому или иному факти-**

**ческому параметру.**

Пример был выше, вот еще пример:



**• Информацию о проведенных изменениях отобразить в отчете на-**

**ряду с информацией, необходимой на предыдущую оценку.**

Выполнено.

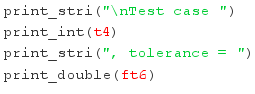
1. **Критерии на 8**

• **Разработанные подпрограммы должны поддерживать многократ-**

**ное использование с различными наборами исходных данных, вклю-**

**чая возможность обработки в качестве параметров различных исходных данных.**

С учетом использования подпрограмм и макросов с разными наборами исходных данных удалось избежать дублирования кода. Например, ввод и вывод строк или целых или дробных чисел осуществляется таким образом:



**• Реализовать автоматизированное тестирование за счет создания**

**дополнительной тестовой программы, осуществляющей про-**

**гон подпрограмм, осуществляющих вычисления для различных те-**

**стовых данных (вместо их ввода). Осуществить прогон тестов обес-**

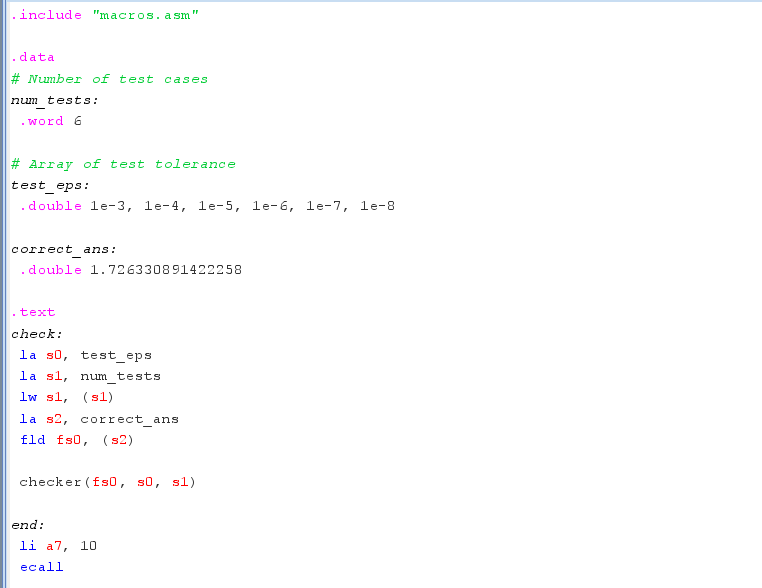
**печивающих покрытие различных ситуаций. В том случае, если ис-**

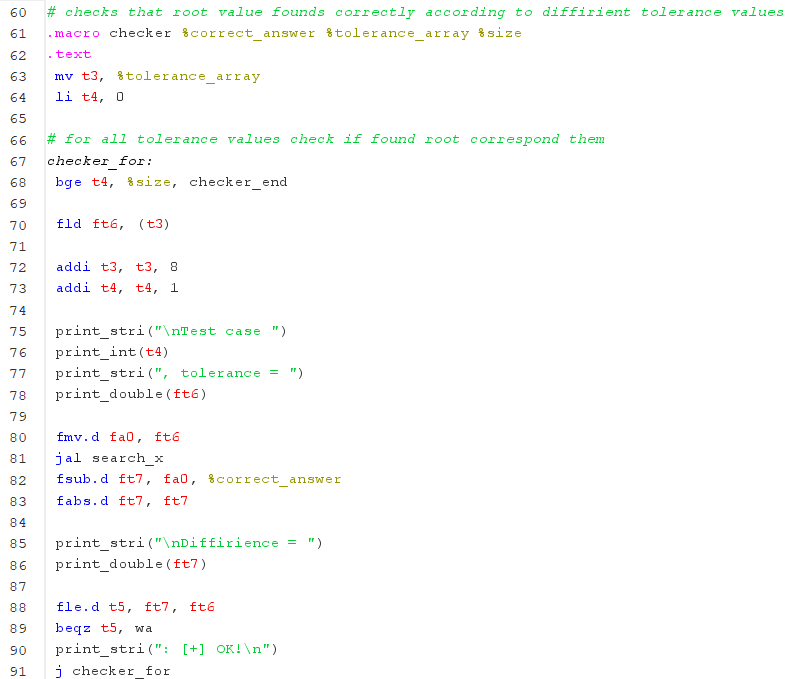
**ходные данные напрямую не прописаны, а точность в условии за-**

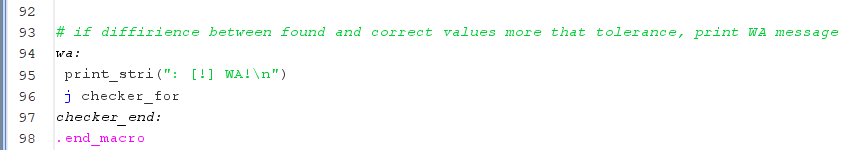
**фиксирована, использовать организацию вычислений с различной**

**точностью.**

Был добавлен файл checker.asm, содержащий программу, осуществляющую автоматическое тестирование на различных наборах входных данных, а также для этого был написан соответствующий макрос.

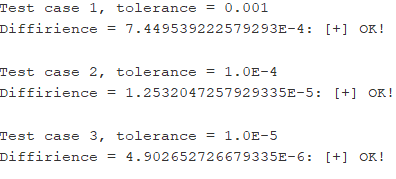


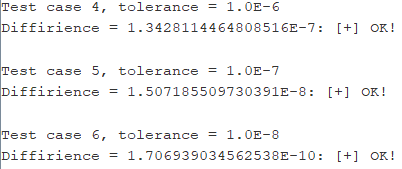




Протестированы различные наборы входных данных:

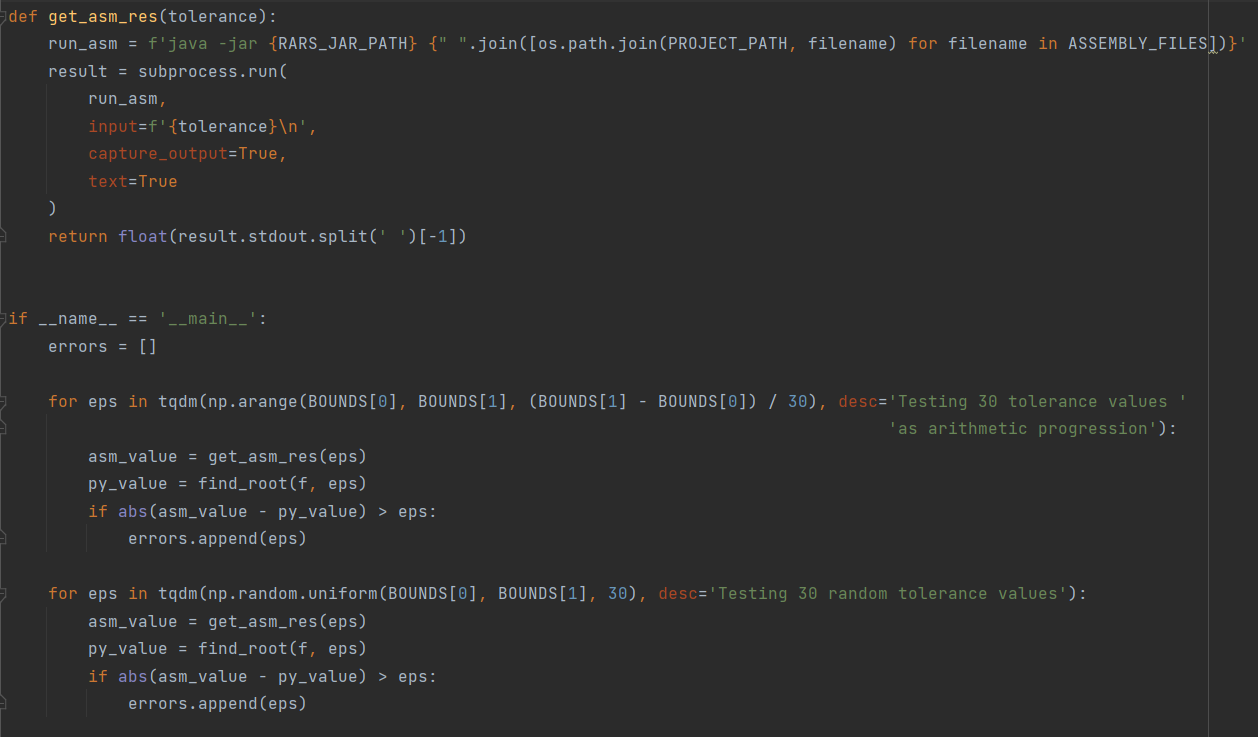
* Минимальный/максимальный размер погрешности
* Все погрешности из диапазона от максимального до минимального значения с шагом уменьшения в 10 раз.



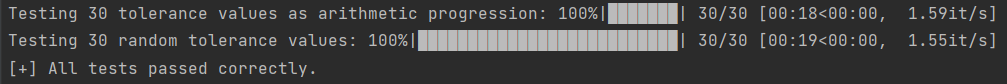


Как видим, все тесты успешно пройдены.

* **Для дополнительной проверки корректности вычислений осуществить аналогичные тестовые прогоны с использованием существующих библиотек и одного из языков программирования высокого уровня по выбору: C, C++, Python.**

Реaлизованы программы на Python: bisect\_serach.py – для решения этой задачи на питоне, а checker.py вызывает программу на risc-v, подает в нее автоматически входные данные и извлекает выходные и сравнивает их с решением bisect\_search.py. Все это происходит на разных входных данных. 

Результат:



**• Добавить информацию о проведенных изменениях в отчет.**

Выполнено.

1. **Критерии на 9 баллов**

**• Добавить в программу использование макросов для реализации**

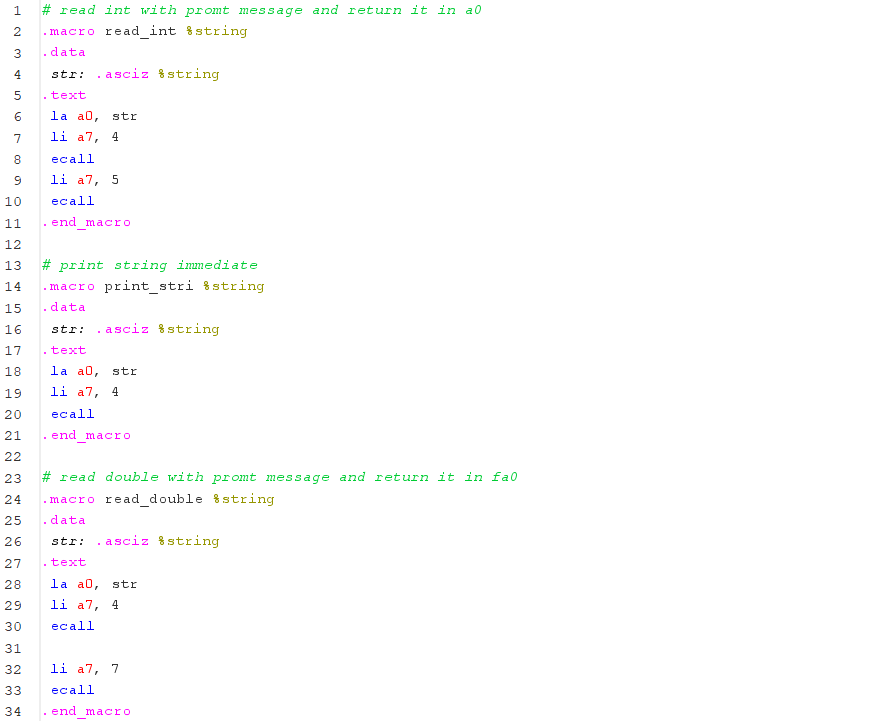
**ввода и вывода данных. Добавить свои макросы, обертывающие**

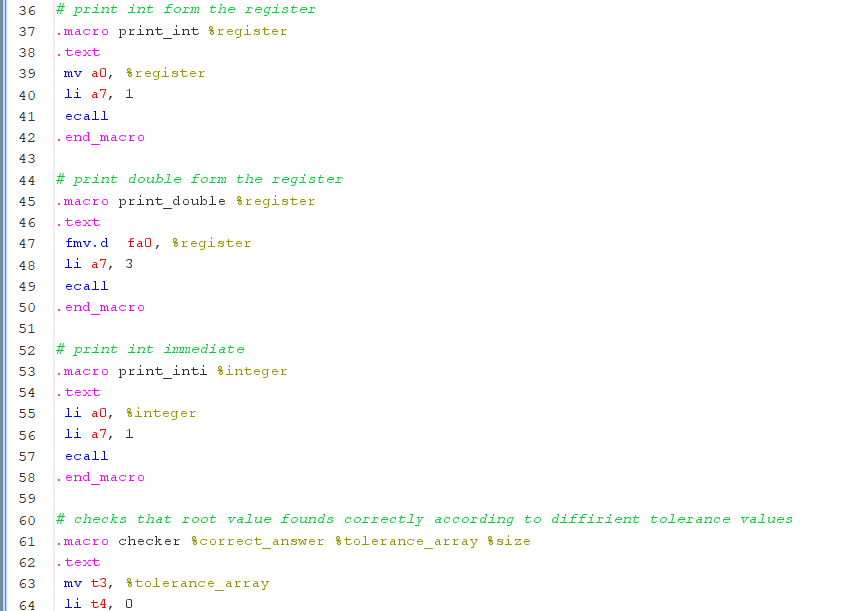
**подпрограммы обработки данных разработанные для решения ос-**

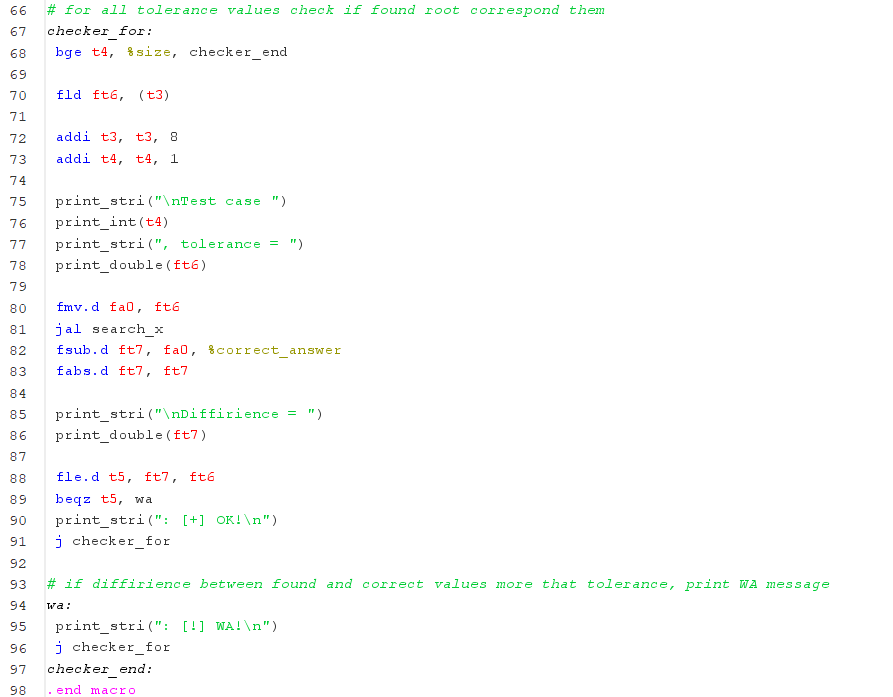
**новной задачи. Макросы должны поддерживать повторное исполь-**

**зование с различными входными и выходными параметрами.**

Макросов добавлено достаточно много. Они содержатся в library.asm. В основном макросы используются для ввода-вывода данных разных типов и тестирования.







1. **Критерии на 10**

**• Программа должна быть разбита на несколько единиц компиля-**

**ции. При этом подпрограммы ввода–вывода должны составлять**

**унифицированные модули, используемые повторно как в програм-**

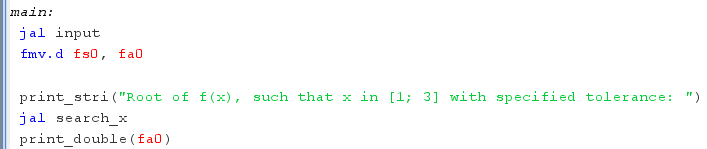
**ме, осуществляющей ввод исходных данных, так и в программе,**

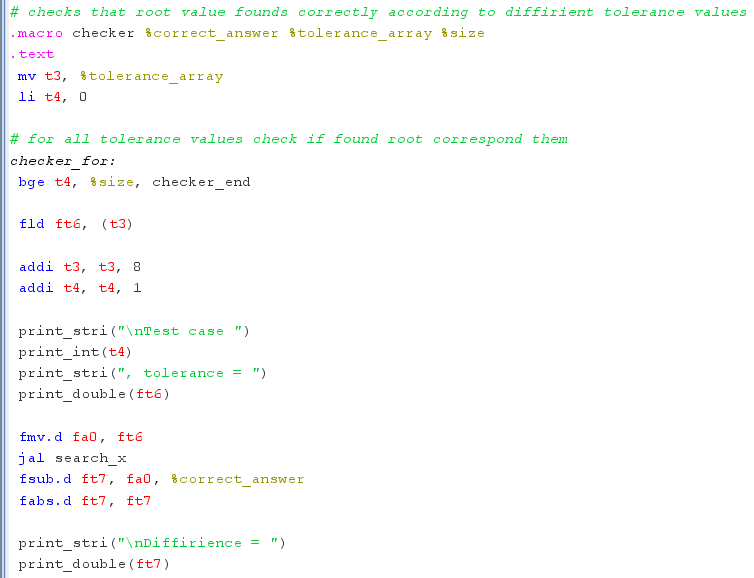
**осуществляющей тестовое покрытие.**

Программа разбита на несколько единиц компиляции:



Подпрограммы ввода-вывода используются в основной программе и в программе реализующей тестирование.





**• Макросы должны быть выделены в отдельную автономную библиотеку**

Как описано выше, все макросы находятся в macros.asm, при чем все они кроме checker являются несобственными.

**• Расширить отчет, дополнив его новыми данными.**

Выполнил.