Выполнила: Муратшина Диана Фанисовна
Биологический факультет, группа ББ-104(2)

ЯЗЫКИ ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ

Содержание

Классификация языков программирования:

- Существующее в мире множество языков программирования классифицируется по четырём основным группам: процедурные, объектно-ориентированные, функциональные и логические. Хоть в этой презентации рассматриваются только вышеперечисленные группы, не стоит забывать о существовании ещё нескольких групп, а именно: аспектно-ориентированные языки, структурные и мультипарадигмальные.

Процедурные языки программирования:

- Вообще, само процедурное программирование представляет собой такое программирование, когда программа отделена от данных и состоит из последовательности команд, обрабатывающих данные. Данные, как правило, хранятся в виде переменных. Весь процесс вычисления сводится к изменению их содержимого. И примерами языков такого программирования являются: Си, Limbo, Паскаль, PureBasic, Maple и многие другие. Процедурное программирование предоставляет возможность программисту определять каждый шаг в процессе решения задачи.

Объектно-ориентированные языки программирования:

- В этих языках переменные и функции группируются в так называемые классы (шаблоны). Благодаря этому достигается более высокий уровень структуризации программы. Объекты, порождённые от классов вызывают методы (функции или процедуры) друг друга и меняют таким образом состояние свойств (переменных). С формально-математической стороны, объектно-ориентированный способ написания программ базируется на процедурной модели программирования, но с содержательной стороны ООП базируется не на функции, а на объекте, как целостной системе, имеющей стандартный автоматический межобъектный интерфейс. Примерами языков такого подхода к программированию являются: C++, Delphi, JavaScript, Python, Swift (который был недавно представлен компанией Apple), PHP и многие другие.

Функциональные языки программирования:

- Краеугольным камнем в парадигме функционального программирования, является функция. Функциональная программа представляет собой набор определений функций. Функции определяются через другие функции или рекурсивно — через самих себя. В процессе выполнения программы функции получают параметры, вычисляют и возвращают результат, в случае необходимости вычисляя значения других функций. Программируя на функциональном языке, программист не должен описывать порядок вычислений. Ему необходимо просто описать желаемый результат в виде системы функций. В функциональном языке при вызове функции с одними и теми же аргументами всегда можно получить одинаковый результат: выходные данные зависят только от входных. Это позволяет средам выполнения программ на функциональных языках кэшировать результаты функций и вызывать их в порядке, не определяемом алгоритмом и распараллеливать их без каких-либо дополнительных действий со стороны программиста. Примерами языков функционального программирования являются: Cat, Dylan, F#, Hope и другие.

Логические языки программирования

- Такие языки ориентированы не на разработку алгоритма решения задачи, а на систематическое и формализованное описание задачи с тем, чтобы решение следовало из составленного описания. В логическом программировании нужно только специфицировать факты, на которых алгоритм основывается, а не определять последовательность шагов, которые требуется выполнить. Логические программы отличаются принципиально низким быстродействием, так как вычисления осуществляются методом проб и ошибок (посредством поиска с возвратами). Логическое программирование основано на теории и аппарате математической логики с использованием математических принципов резолюций. Примерами языков такого программирования являются: Prolog, Mercury, Planner, Visual Prolog, OZ и несколько других.

Технологический цикл разработки программ:

- Говоря о техническом цикле разработки программ, стоит упомянуть такое понятие, как среда программирования.

Среда программирования – это совокупность программ, обеспечивающих технологический цикл разработки программ: анализ, спецификация, проектирование, кодирование (редактирование, компиляция, компоновка), тестирование, отладка.

В базовые компоненты среды программирования входят:

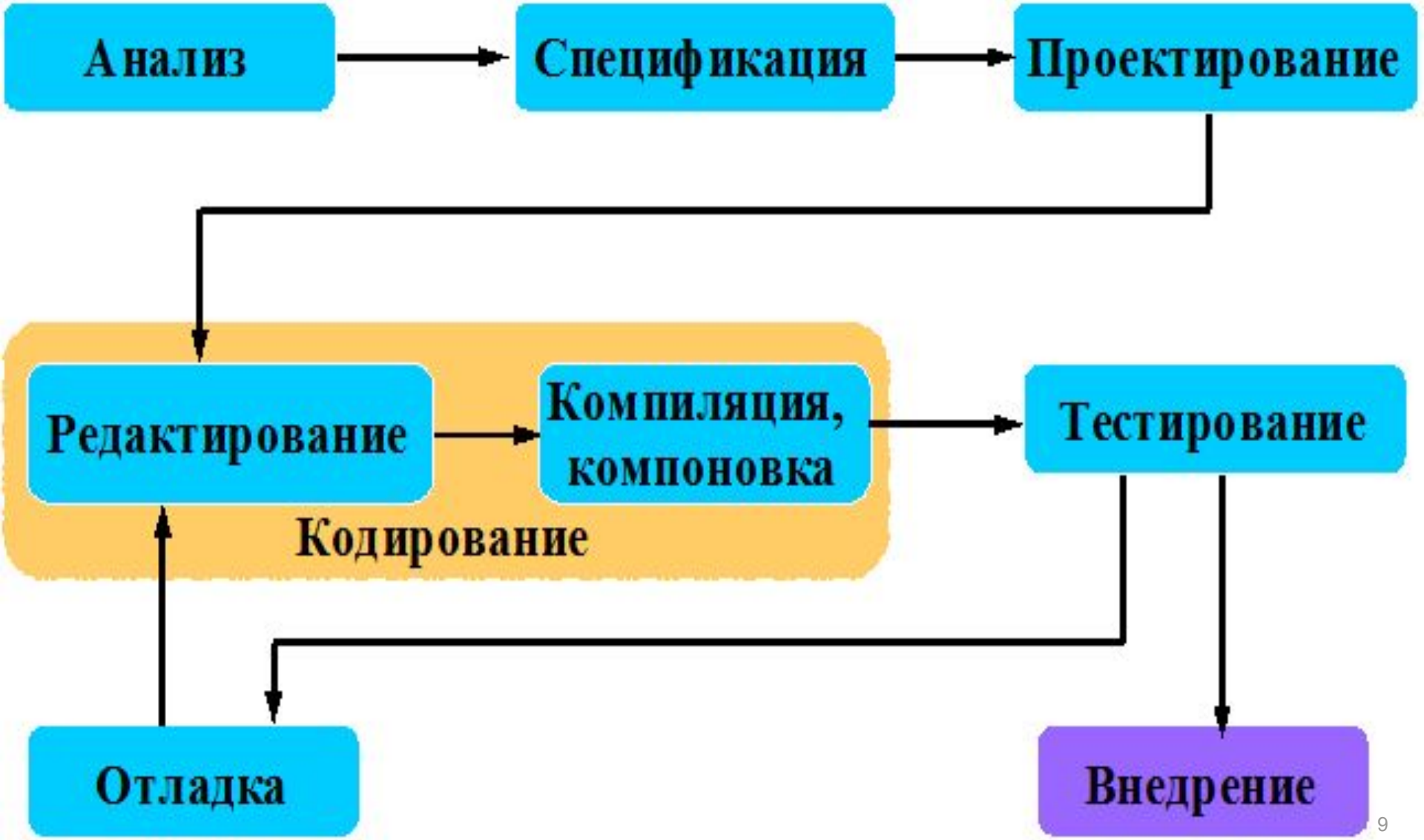
1) Редактор – средство создания и изменения исходных файлов с текстом программы.

2) Компилятор – транслирует исходный файл в объектный файл, содержащий команды в машинном коде для конкретного компьютера.

3) Компоновщик (редактор связей) – собирает объектные файлы программы и формирует исполняемый файл (разрешая внешние ссылки между объектными файлами).

4) Отладчик – средство управления выполнением исполняемого файла на уровне отдельных операторов программы для диагностики ошибок.

Технологический цикл разработки программ:



Редактирование:

- Редактирование в плане программирования – это практически неотъемлемый процесс при написании программы. Чаще всего он происходит во время самого написания исходного кода. Редактирование может относится не только к исправлениям мелких ошибок (тем не менее, значительных для кода), но и к исправлениям больших кусков программного кода. Существуют некоторые редакторы кода, которыми чаще всего выступают онлайн-сервисы, способные находить ошибки и исправлять их. Но, так или иначе, их возможности редактирования ограниченны в отличии от возможностей грамотного программиста. Современные компиляторы позволяют лучше находить ошибки в коде, помогая, например, подсветкой синтаксиса или всевозможными предупреждениями.

Компиляция:

- Термин происходит от английского “compilation” — объединение, составление.
- Это, своего рода, есть преобразование программой-компилятором исходного текста программы, написанного на языке высокого уровня в машинный язык, в язык, близкий к машинному, или в объектный модуль. Компиляция относится к обработке файлов исходного кода и создании объектных файлов проекта. На этом этапе не создается исполняемый файл. Вместо этого компилятор просто транслирует высокоуровневый код в машинный язык.

Отладка:

- Отладка – это этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки.

  Чаще всего, при написании сложных программ, процесс отладки может очень сильно затянуться по времени, несмотря на существование специальных программ, которые могут автоматиризорвать этот процесс. Программист обязательно должен принять участи в отладке программы.

Методы отладки программ.

- Для локализации и установки точной природы ошибки в программе используют статические и динамические методы отладки программ.

  К статическим методам относятся методы отладки, при которых не требуется выполнение отлаживаемой программы на ЭВМ. Они обычно требуют больших усилий от программиста и незначительных затрат машинного времени. Они универсальны и пригодны для отладки программ, написанных на любом языке программирования и на любой ЭВМ.
- Статические методы включают:

  - ручную прокрутку программы;
  - прокрутку программы программными анализаторами;
  - коллективную проверку программ;
  - проверку программы программистом-технологом.

Методы отладки программ:

- Динамические методы связаны со значительным расходом машинного времени и, возможно, не меньшими затратами труда программиста. В этом случае отладка программ происходит совместно с их выполнением на ЭВМ. Динамические методы отладки программ, как правило, привязаны к конкретной ЭВМ и к конкретному транслятору (компилятору).
- К динамическим методам относятся:

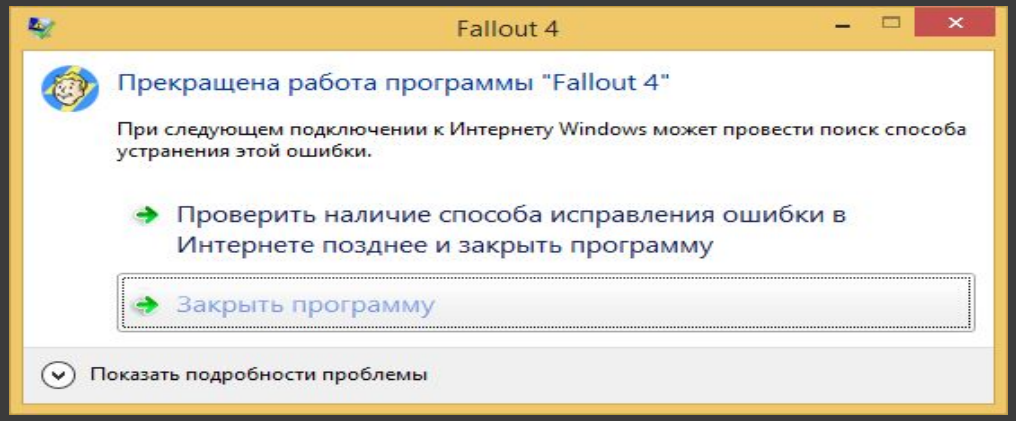
- тестирование;
- поиск ошибок с использованием системных средств;
- отладка программы в интерактивном режиме.

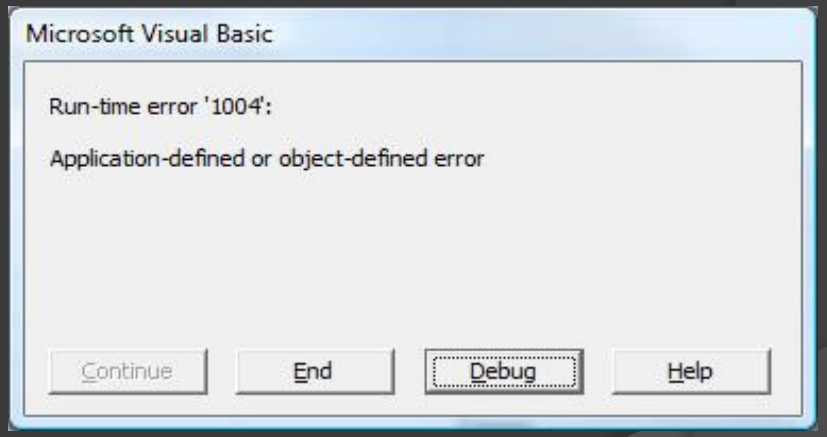
Виды ошибок:

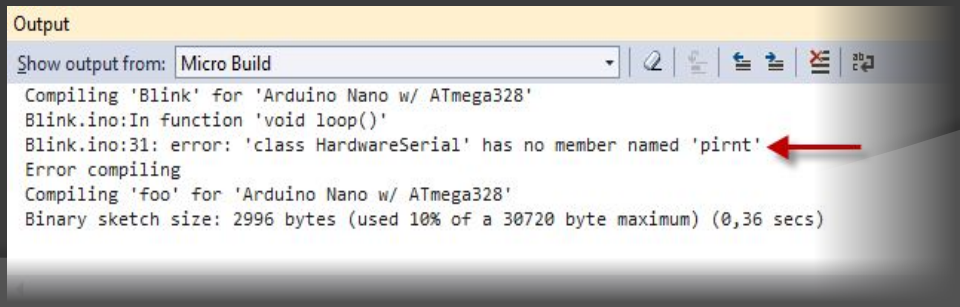
- Ошибки делают все, начинающие и даже самые искусные программисты, это лишь дело времени. Чаще всего ошибки появляются из-за не внимательности, где-то запятую пропустили или не поставили скобочку и т.д. Но даже такие банальные ошибки необходимо отлавливать и исправлять.
- Существует три основных типа программных ошибок: ошибки времени компиляции, ошибки времени выполнения и логические ошибки.
- Ошибки компиляции или синтаксические ошибки встречаются, когда забывают объявить переменную, передают ошибочное количество параметров процедуры, при назначении действительного значения целочисленной переменной. Это означает, что записываются операторы, которые не согласуются с правилами языка программирования.
- Другой тип ошибок - ошибки времени выполнения программы или семантические ошибки. Они встречаются, когда пользователь компилирует синтаксически корректную программу, которая пытается сделать что-нибудь запрещенное во время ее выполнения, например, открывает несуществующий файл для ввода или производит деление на 0.

Виды ошибок:

- Программа пользователя может содержать и логические ошибки. Это означает, что программа делает то, что ей указали вместо того, что хотелось бы. Может отсутствовать инициализация переменной; могут оказаться ошибочными вычисления; рисунки, изображенные на экране, выглядят неправильно; программа может просто работать не так, как было задумано. Такие ошибки находятся с большим трудом, и тут лучше использовать, интегрированный в компилятор, отладчик.





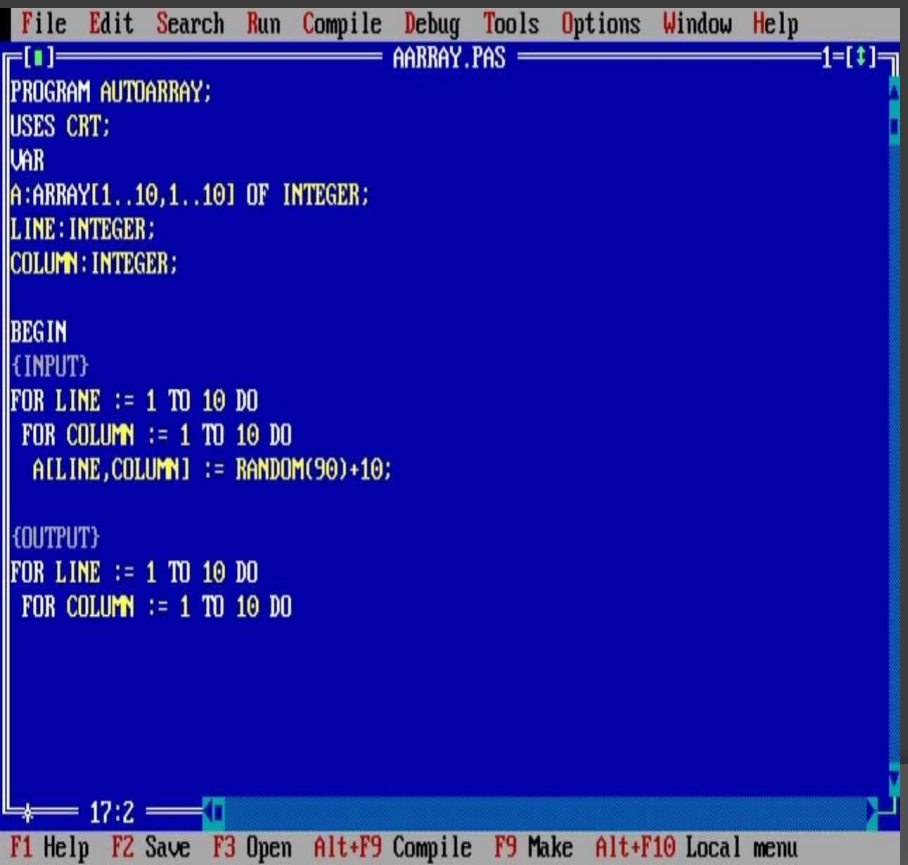


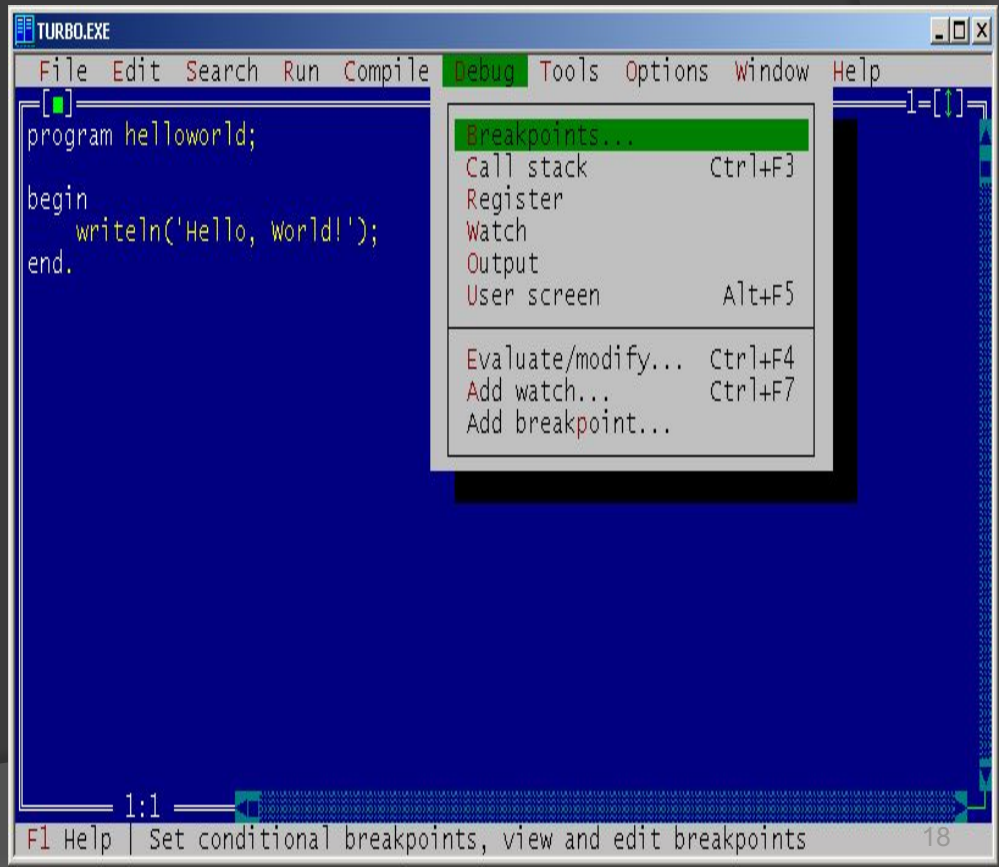
Язык программирования Паскаль:

- Язык Паскаль был разработан в 1970 г. Никлаусом Виртом, как язык, обеспечивающий строгую типизацию и интуитивно понятный синтаксис. Он был назван в честь французского математика, физика и философа Блеза Паскаля. До сих пор Паскаль заслуженно считается одним из лучших языков для начального обучения программированию. Его современные модификации, такие как Object Pascal, широко используются в промышленном программировании.
- Тем не менее, первоначально язык обладал множеством недостатков: невозможность передачи функциям массивов переменной длины, отсутствие нормальных средств работы с динамической памятью, ограниченная библиотека ввода-вывода, отсутствие средств для подключения функций написанных на других языках, отсутствие средств раздельной компиляции и т. п.
- Однако многие недостатки языка не проявляются или даже становятся достоинствами при обучении программированию. К 1980-м годам Паскаль стал основой для многочисленных учебных программ, в отдельных случаях на его основе были созданы специализированные обучающие языки программирования.

Язык программирования Паскаль:

- Наиболее известной реализацией Паскаля, обеспечившей широкое распространение и развитие языка, является Turbo Pascal фирмы Borland.

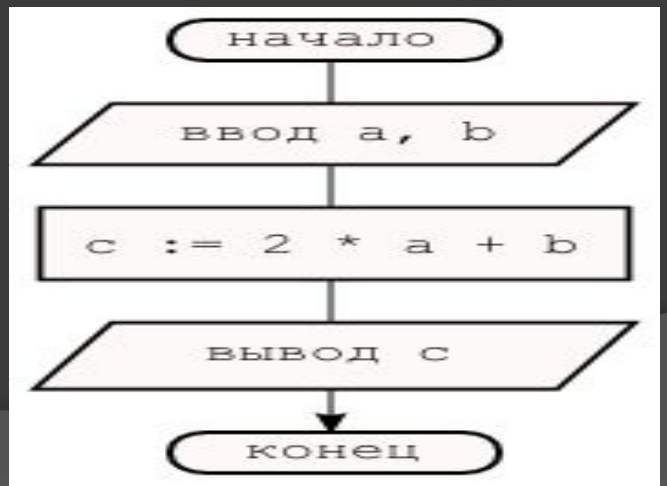




Никлаус Вирт

Программы линейной структуры:

- Для дачи определения такой программы необходимо обратиться к такому понятию, как Алгоритм линейной структуры.
- Алгоритм линейной структуры - это алгоритм, в котором блоки выполняются в указанном порядке, последовательно друг за другом.
- Значит, что программа линейной структуры реализует соответствующий линейный алгоритм. Иначе говоря, программа имеет линейную структуру, если все операторы (команды) выполняются последовательно друг за другом.
- Для примера можно рассмотреть простой алгоритм подобной программы, где: на ввод даются два числа (a, b); переменной «c» присваивается значение суммы удвоенного числа «a» и числа «b»; затем, осуществляется вывод полученного значения «c».

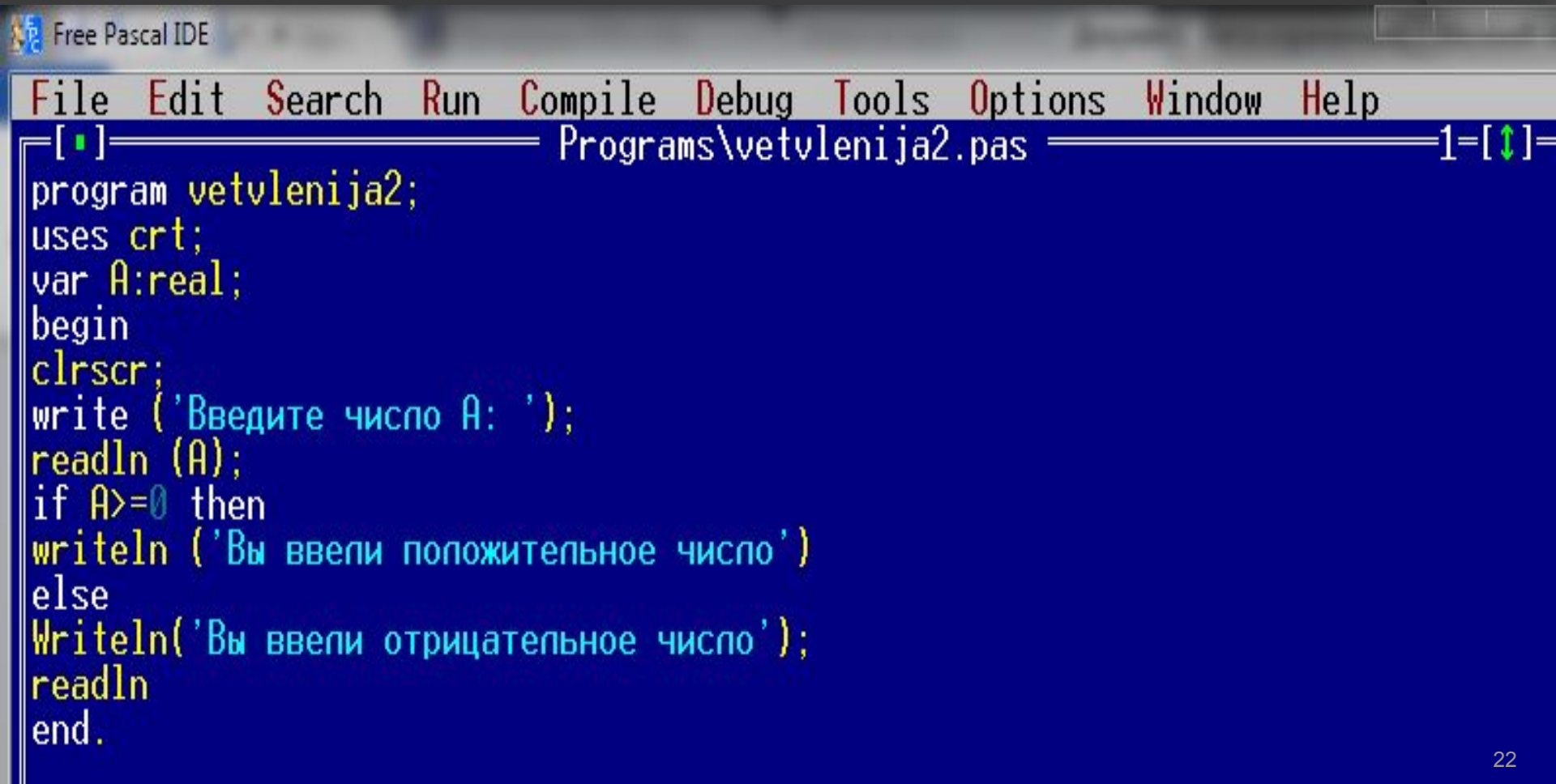


Операторы ветвления и операторы цикла:

- Оператор ветвления - это конструкция языка программирования, обеспечивающая выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд (наборов команд) в зависимости от значения некоторого выражения.
- Оператор ветвления применяется в случаях, когда выполнение или невыполнение некоторого набора команд должно зависеть от выполнения или невыполнения некоторого условия. Ветвление — одна из трёх (наряду с последовательным исполнением команд и циклом) базовых конструкций структурного программирования.
- В языке Паскаль, например, для использования ветвления существует конструкция IF...THEN или ELSE.

Операторы ветвления и операторы цикла:

- А на этом скриншоте окна программы можно увидеть, как синтаксически правильно можно применять оператор ветвления в языке Паскаль с помощью конструкции if…else:



Операторы ветвления и операторы цикла:

- Если в программе возникает необходимость неоднократного выполнения некоторых операторов, то используются операторы повтора (цикла). Например, в языке Паскаль различают три вида операторов цикла: while, repeat, for.
- Если число повторений оператора (составного оператора) заранее неизвестно, а задано лишь условие его повторения (или окончания), используются операторы while, repeat. Оператор for используется, если число повторений заранее известно.

Операторы ветвления и операторы цикла:

- Тут, на картинке, можно увидеть, как правильно использовать оператор цикла в языке Паскаль с помощью конструкции while…do:

File Edit Search Run Compile Debug Tools Options Window Help

Programs\ciklih3.pas

```
program ciklih3;
uses crt;
Var S:integer;
    N:Integer;
Begin
clrscr;
S:=0;
N:=1;
While N<=50 Do
Begin
S:=S+N;
N:=N+1;
End;
Writeln ('S=',S);
readln
End.
```

Список литературы:

- Джозеф Джарратано «Экспертные системы: принципы разработки и программирование»
- *Вольфенгаген В. Э.* Конструкции языков программирования. Приёмы описания.
- Некоторая часть информации для презентации была взята с сайтов: cyberforum.ru, studopedia.ru, mojainformatika.ru, cppstudio.com и wikipedia.org.