Курс «Основы программирования»

Федорук Елена Владимировна ст. преподаватель каф РК-6 МГТУ им.Н.Э.Баумана

Лекция №1

Стадии создания программного продукта

При разработке программного продукта можно выделить следующие стадии:

1. **Стадия предпроектных исследований и технического задания** (постановка задачи) — определение требований к программному продукту и осуществление формальной постановки задачи;
2. **Стадия технического предложения** (анализ) — определение методов решения задачи;
3. **Стадия эскизного проектирования**(проектирование) — разработка структуры программного продукта, выбор структур для хранения данных, построение и оценка алгоритмов подпрограмм и определение особенностей взаимодействия программы с вычислительной средой (другими программами, операционной системой и техническими средствами);
4. **Стадия технического проектирования** (программирование) — составление программы на выбранном языке программирования, ее отладка и тестирование;
5. **Стадия рабочего проектирования** — оформление документации;
6. **Стадии испытаний и внедрения в эксплуатацию** — всестороннее тестирование программы и сопровождение при внедрении в эксплуатацию.

При разработке сложных программных продуктов обычно используют одну из двух технологий: *структурное программирование* и *объектно-ориентированное программирование*.

Первая технология рекомендует разбивать (декомпозировать) сложную программу на подпрограммы (процедуры, функции), решающие отдельные подзадачи, т.е. базируется на процедурной декомпозиции.

Вторая технология использует подход, при котором в предметной области задачи выделяются отдельно функционирующие элементы. Поведение этих объектов программно моделируется с использованием специальных средств, а затем, из готовых объектов собирается сложная программа. Таким образом, в основе второй технологии лежит объектная декомпозиция.

Стадия предпроектных исследований и технического задания

Часто *стадию предпроектных исследований и технического задания* называют стадией постановки задачи. Постановка задачи должна содержать достаточно информации для того, чтобы позволить программисту или аналитику однозначно определить, что будет делать создаваемая

программа. Детали того, как она будет это делать, должны определиться позже, при разработке алгоритма. Для того, чтобы грамотно выполнить постановку задачи, необходимо произвести предпроектные исследования предметной области, определить, существуют ли аналоги будущего программного продукта, их достоинства, недостатки и т.д.

Постановка решаемой на компьютере задачи должна включать список ее входных данных, список требуемых результатов и любые инструкции (правила), которых нужно следовать при решении задачи. В постановке также должно быть указано, должна ли задача решаться для конкретного случая (конкретных данных) или должна обобщаться для переменных входных данных. Если возможно, решение должно обобщаться так, чтобы его можно было использовать для решения класса задач. Например, программа, написанная для вычисления текущей зарплаты, должна быть написана так, чтобы она могла бы периодически использоваться и в будущем без изменений.

В результате согласования между заказчиком и исполнителем всех перечисленных вопросов составляют *техническое задание* (ТЗ) в соответствии с ГОСТ 19.201-78, которое служит основанием для дальнейшей работы.

Стадия технического предложения

На *стадии технического предложения* выполняется *анализ* задачи – это определение и детализация логического порядка действий, которые нужно выполнить над исходными данными, чтобы получить требуемое решение. На этом этапе процесса решения задачи следует в общих чертах описать, что необходимо сделать. Детали того, как это должно быть сделано, будут уточняться на следующем шаге.

Пример 1

Найти корни квадратного уравнения с заданными коэффициентами.

Необходимо определить метод решения задачи, будет ли программа находить общее решение или только действительные корни.

Часто формальная постановка задачи однозначно определяет метод ее решения. Если задача может быть решена несколькими методами, выбирается один из них с учетом сложности реализации, точности результатов и т.д.

Стадия эскизного проектирования

На *стадии эскизного проектирования* при использовании процедурного подхода сложные задачи разбиваются на подзадачи, для которых может строиться своя модель и выбираться свой метод решения. При этом результаты одной подзадачи могут использоваться как исходные данные в другой.

Целесообразно проверить правильность выбранных моделей и методов, выполнив их вручную для некоторых значений исходных данных.

При определении типов исходных данных необходимо также продумать, для каких сочетаний этих данных результат не существует или не может

быть получен данным методом, что также надо учитывать при разработке программы.

Одновременно с написанием алгоритма, необходимо точно определить тип и структуру обработанных данных. В одних случаях данными могут быть несколько обычных чисел, в других организация данных будет более сложной.

При определении структуры данных с каждым объектом данных должно быть связано осмысленное имя или идентификатор. При разработке программы идентификаторы будут связаны с расположением данных в памяти.

На данной стадии разрабатываются и оцениваются алгоритмы подпрограмм.

Алгоритмом называют формально описанную последовательность действий, которые необходимо выполнить для получения требуемого результата.

Разработка алгоритма состоит в пошаговом описании предлагаемого решения задачи. Каждый шаг должен быть описан в виде кратких и точных операторов с использованием структурированного языка или псевдокода.

Каждой процедуре должно быть дано содержательное имя или идентификатор, для того, чтобы ее можно было вызвать по имени как модуль из другой процедуры:

Пример 1

`find_area` (Найти площадь),
`calc_balance` (Вычислить_баланс)
`prepare_fit` (Подготовить_декларацию)

В операторах разработчик алгоритма должен использовать идентификаторы предварительно определенных данных. Данные, определенные вне модуля (при постановке задачи или ее анализе), рассматриваются как внешние данные и должны быть включены в разработанное определение структуры данных. Внешние данные могут совместно использоваться несколькими процедурами. Данные, принадлежащие одному модулю, являются внутренними или временными и могут быть определены внутри самого модуля. Использование внутренних (или локальных) данных повышает модульность программы.

Алгоритм должен быть детализирован до такой степени, когда каждый оператор выполняет простую операцию или вызов другой процедуры. Этого можно достичь путем использования повелительных и условных операторов (предложений). Разработка алгоритма завершается, когда каждый из операторов может быть записан непосредственно на языке программирования.

Сравните два алгоритма.

Пример 2

- Алгоритм 1
- Quad_roots_1 (квадр_корни_1)
  - Ввести три коэффициента квадратного уравнения
  - Вывести корни

Пример 3

- Алгоритм 2
- Quad_roots_ 2 (квадр_корни_2)
- Замечание – для случая комплексных корней алгоритм не применим.
  - Ввести три коэффициента квадратного уравнения a, b, c.
  - Вычислить

$$\text{корень}_1 = \frac{-b + \sqrt{(b^2 - 4 \cdot a \cdot c)}}{2 \cdot a}$$

  - Вычислить

$$\text{корень}_2 = \frac{-b - \sqrt{(b^2 - 4 \cdot a \cdot c)}}{2 \cdot a}$$

  - Напечатать значения корень_1, корень_2

Какой из алгоритмов лучше? Алгоритм 1 очень "грубо" описывает требуемую последовательность, алгоритм 2 представляет частный случай решения задачи. Для решения задачи в общем виде необходимо добавить в алгоритм расчет мнимых корней.

На изображение схем алгоритмов существуют ГОСТ 19.701-90, согласно которому каждой группе действий ставится в соответствие блок особой формы.

Стадия технического проектирования

На *стадии технического проектирования* разработанные алгоритмы реализуют, составляя по ним текст программы с использованием конкретного языка программирования. Язык может быть определен в техническом задании, а может выбираться исходя из особенностей конкретной разработки.

Возможно будет выбран язык программирования Си. Программа должна быть занесена в исходный (текстовый) файл на вашем компьютере с использованием стандартного редактора. Файл, содержащий исходный текст программы на языке Си, должен иметь расширение **.c**.

Далее программу необходимо перевести в последовательность машинных команд (машинный код). Для этого используется специальная программа — компилятор.

Шаги компиляции Си

Самые простые ошибки в программе — это ошибки компиляции. Для исправления этих ошибок обычно достаточно внимательно изучить сообщение об ошибке и соответствующий текст программы, внести изменения в программу и повторно выполнить компиляцию. Следующая группа ошибок — ошибки компоновки. Для исправления таких ошибок необходимо сверить объявления, определения и вызовы функций, проверить правильность использования внешних переменных и стандартных функций.

Если исходный текст программы не содержит ошибок, то компилятор создает исполнимый код программы. Далее программа выполняется. При этом необходимо выявить ошибки выполнения. Для исправления таких ошибок может потребоваться их локализация, т.е. уточнение, при выполнении какого фрагмента программы происходит нарушение нормального вычислительного процесса.

Процесс локализации и исправления ошибок получил название *отладки программы*. При отладке программы часто используют специальные программы – отладчики, которые позволяют выполнить любой фрагмент программы в пошаговом режиме и проверить содержимое интересующих нас переменных.

Отлаженная программа подвергается тестированию. *Тестирование* — это процесс выполнения программы при различных наборах данных с целью обнаружения логических ошибок. Для поиска логических ошибок также можно использовать отладчик: по шагам отследить процесс получения результатов. Однако полезно бывает выполнить программу вручную, фиксируя выполнение команд на бумаге. При этом поможет пример расчета, выполненный вручную на этапе анализа и выбора метода.

Стадия рабочего проектирования

На *стадии рабочего проектирования* оформляется программная документация.

Если программу предполагается использовать и сопровождать в течении какого-либо срока, она должна быть документирована. Несмотря на то, что здесь документирование рассматривается как отдельный этап в процессе разработки решения задачи, оно должно выполняться на протяжении всего этого процесса.

Документация должна включать:

- описание постановки задачи;
- описание анализа задачи;
- описание определения структуры данных;
- описание алгоритма;
- текст программы с комментариями;
- тесты с входными и выходными данными;

Стадия испытаний

На *стадии испытаний* выполняется тестирование программы. Тестирование выполняется постоянно на протяжении всей работы над программой. Завершающее тестирование должно быть всесторонним. При этом должны быть учтены все возможные варианты входных данных, а также все ошибочные ситуации.

Стадия внедрения в эксплуатацию

После завершения разработки программы ее использование должно контролироваться на *стадии внедрения в эксплуатацию* с целью наблюдения за правильностью ее работы и установления обратной связи с конечным пользователем, который может внести предложения по повышению эффективности и простоты использования программы.