|  |  |  |
| --- | --- | --- |
| Дата | Содержание выполняемой работы | Подпись руководите ля |
| 20.05.2022,  21.05.2022 | **Практическая работа №21-22**  **Тема:** Обоснование выбора средств реализации проекта. Кодирование модулей разрабатываемого программного обеспечения.  **Наименование работы:** Кодирование основных модулей программной системы. Кодирование дополнительных модулей программной системы.  **Цель:** Закрепление умений и навыков по кодированию модулей программного средства, корректировке кода готового ПС.  Часть 1.  **Задание 1.**  Изучил теоретический материал «Методы программирования.pdf». Законспектировал методы и их характеристики в дневнике-отчете.  В основе того или иного языка программирования лежит некая руководящая идея, вызванная потребностями или, чаще всего, кризисом в области программирования и создания программного обеспечения, которая оказывает существенное влияние на стиль программирования и помогает преодолеть указанный кризис . Рассмотрим вкратце историю появления и развития основных стилей программирования и процедурных алгоритмических языков.  Машинно-ориентированное программирование появилось одновременно с созданием электронных вычислительных машин.  Процедурное программирование. Основная идея этого стиля –алгоритмизация процесса решения задачи и выбор наилучшего алгоритма (по расходу памяти или по быстродействию).  Модульное программирование. Здесь основная идея заключалась в том, чтобы "спрятать" данные и процедуры внутри независимых программных единиц -модулей.  Основные характеристики программного модуля. Не всякий программный модуль способствует упрощению программы. Выделить хороший с этой точки зрения модуль является серьезной творческой задачей. Для оценки приемлемости выделенного модуля используются некоторые критерии.  Майерс предлагает для оценки приемлемости программного модуля использовать более конструктивные его характеристики:  размер модуля;  прочность модуля;  сцепление с другими модулями;  рутинность модуля (независимость от предыстории обращений к нему).  Размер модуля измеряется числом содержащихся в нем операторов или строк. Модуль не должен быть слишком маленьким или слишком большим.  Прочность модуля-это мера его внутренних связей. Чем выше прочность модуля, тем больше связей он может спрятать от внешней по отношению к нему части программы и, следовательно, тем больший вклад в упрощение программы он может внести.  Сцепление модуля-это мера его зависимости по данным от других модулей. Характеризуется способом передачи данных. Чем слабее сцепление модуля с другими модулями, тем сильнее его независимость от других модулей.  Рутинность модуля-это его независимость от предыстории обращений к нему. Модуль называется рутинным, если результат (эффект) обращения к нему зависит только от значений его параметров (и не зависит от предыстории обращений к нему). Модуль называется зависящим от предыстории, если результат (эффект) обращения к нему зависит от внутреннего состояния этого модуля, изменяемого в результате предыдущих обращений к нему.  Модульное проектирование относится к процессу расчленения больших проблем на более узкие, более управляемые под проблемы. Первым шагом проектирования является решение, в каком месте должна быть граница между этими под проблемами.  При обращении к проблеме на стадии проектирования первым альтернативным выбором должна быть функциональная декомпозиция, т.е. разбиение проблемы на более узкие, вполне поддающиеся управлению функциональные единицы, где каждая единица выполняет завершенную, легко идентифицируемую задачу.  Иногда обнаруживается, что после определения модулей программы создано нечто громоздкое и неуклюжее. Это часто случается тогда, когда модули при выполнении возложенных на них задач требуют доступа к обширному количеству данных. Чаще всего это легко может произойти, если на модуль возложено выполнение нескольких опций.  Как уже отмечалось выше, в качестве модульной структуры программы принято использовать древовидную структуру, включая деревья со сросшимися ветвями. В узлах такого дерева размещаются программные модули, а направленные дуги (стрелки) показывают статическую подчиненность модулей, т.е. каждая дуга показывает, что в тексте модуля, из которого она исходит, имеется ссылка на модуль, в который она входит.  Особенностью рассмотренных методов восходящей и нисходящей разработок является требование, чтобы модульная структура программы была разработана до начала программирования (кодирования) модулей. Разработка модульной структуры программы и ее кодирование производятся на разных этапах разработки программного средства (ПС): первая завершает этап конструирования ПС, а второе -открывает этап кодирования. Однако эти методы вызывают ряд возражений: представляется сомнительным, чтобы до программирования модулей можно было разработать структуру программы достаточно точно и содержательно. На самом деле это делать не обязательно. Ниже описываются конструктивный и архитектурный подходы к разработке программ [3], в которых модульная структура формируется в процессе программирования (кодирования) модулей.  Аналогичные действия производятся при программировании любого другого модуля, который выбирается из текущего состояния дерева программы из числа специфицированных, но пока еще не запрограммированных модулей. В результате этого производится очередное деформирование дерева программы.  **Задание 2.**  Разработка и кодирование пользовательского меню. Скриншоты меню, а также листинг кода вставил в Приложении 21.1.  **Задание 3.**  Кодирование пользовательского интерфейса. Скриншоты разработанных форм и листинг код вставил в Приложение 21.2.  Часть 2.  **Задание 1.**  Разработка оставшихся форм приложения, их кодирование. Скриншоты разработанных форм и листинг код вставил в Приложение 22.1.  **Задание 2.**  Проверил совместимость модулей программного средства. При выявлении ошибок исправил их. Сделал вывод о готовности моего ПС. |  |