МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тульский государственный университет»

Кафедра вычислительной техники

# Исследование вычислительной эффективности объектно-ориентированных приложений

## МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ

по дисциплине

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Направление подготовки: 09.03.01«Информатика и вычислительная техника»

Профиль подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем»

Форма обучения: очная

Тула, 2016

Содержание

## 2. Введение

В настоящее время объектно-ориентированное программирование (ООП) является доминирующим стилем при создании больших программ и программных систем. Процедурно-ориентированное программирование, широко использовавшееся до появления ООП, обычно позволяет создавать более эффективные в вычислительном отношении реализации приложений, что является существенным фактором при разработке систем реального времени. На практике эти два стиля программирования часто используются совместно, позволяя варьировать степень их применения в программах.

Использование объектно-ориентированного (ОО) подхода при разработке программного обеспечения (ПО) позволяет преодолеть естественную сложность разрабатываемого ПО, упростить процесс отладки и последующего сопровождения, расширения и переноса ПО на другие платформы.

ОО подход включает в себя объектно-ориентированный анализ (ООА), дизайн (проектирование) (ООД) и программирование.

Объектно-ориентированный анализ – это методология, при которой требования к системе воспринимаются с точки зрения классов и объектов, выявленных в предметной области.

Объектно-ориентированное проектирование – это методология проектирования, соединяющая в себе процесс объектной декомпозиции и приемы представления логической и физической, статической и динамической моделей проектируемой системы.

Объектно-ориентированное программирование – это методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования. Идеальное ОО приложение должно быть расширяемым, масштабируемым, сопровождаемым и переносимым. Расширяемость означает простоту добавления новых функций, сопровождаемость (поддерживаемость) является показателем простоты поиска и устранения ошибок, а переносимость означает простоту переноса программы в другую операционную систему или ее новую версию. Масштабируемость определяет способность приложения работать при увеличении нагрузки, используя предусмотренные для этого средства.

На результатах ООА формируются модели, на которых основывается ООД, а ООД, в свою очередь, создает фундамент для окончательной реализации системы с использованием методологии ООП.

Процесс разработки ОО программного обеспечения представляет собой итеративный процесс, использующий ООА, ООД и ООП, в котором возможно многократное возвращение на предыдущие этапы разработки.

Основным понятием ООП является класс. *Класс* (class) определяет группу объектов с общими свойствами (атрибутами), поведением (функциями), семантикой и связями с другими объектами. Класс можно трактовать как шаблон для создания объектов. Каждый объект является экземпляром некоторого

класса, причем только одного. Класс может наследовать один или нескольких интерфейсов, реализуя свойства, события и методы каждого из них.

Основными концепциями ООП, которыми руководствуются при создании классов, являются инкапсуляция, наследование и полиморфизм (параметрический и основной, применяемый при наследовании). При создании сложных объектов наряду с наследованием (отношением «is-a») широко используется включение объектов (отношение «is-part-of»)).

Разработка ПО обычно производится с помощью специальных CASE-средств для автоматизированного проектирования и создания программ (computer-aided software engineering - CASE). Накопленный опыт автоматизированного создания программных систем показал, что одной из основных целей разрабатываемых автоматизированных процессов создания и эксплуатации ПО является стремление уменьшить зависимость проектной организации от конкретных исполнителей. Достигается это в первую очередь высокой дисциплинированностью и хорошей документированностью самого процесса проектирования ПО. Появление языка UML значительно способствовало решению этой задачи.

Существует огромное количество методологий и рекомендаций, направленных на повышение эффективности процесса проектирования программных систем. Среди них можно выделить принципы SOLID для гибкого проектирования объектно-ориентированного ПО и такие известные методики как RUP (Rational Unified Process), XP (eXtreme Programming), MSF (Microsoft Solution Framework) и др. Однако, в основе всех этих методологий лежит универсальная методика - принцип повторного использования, представленный техникой шаблонного проектирования. Понятие "шаблон (паттерн) проектирования" – это описание взаимодействия объектов и классов, адаптированных для решения общей задачи проектирования в конкретном контексте.

В данной курсовой работе создаются процедурно-ориентированная и объектно-ориентированная реализации конкретного приложения и проводится сравнительный анализ их вычислительной эффективности.

## 2. Цель и задачи выполнения курсовой работы

Курсовая работа "Исследование вычислительной эффективности объектно-ориентированных приложений" выполняется для закрепления знаний по курсу "Объектно-ориентированное программирование" и приобретения навыков объектно-ориентированной и процедурной реализаций прикладной задачи (задачи вычисления площади геометрической фигуры методом Монте Карло) с использованием различных языков, инструментальных систем и библиотек, автоматизирующих проектирование, программирование и отладку создаваемых приложений.

Задачами курсовой работы являются:

- приобретение навыков решения вычислительных задач;
- практическое освоение современных инструментальных систем разработки ПО;
- сравнительный анализ вычислительной эффективности процедурных и объектно-ориентированных программ;
- получение навыков создания управляемых и неуправляемых программ на языке C++ платформы Microsoft .NET Framework;
- приобретение практических навыков оформления и выпуска документации в соответствии с требованиями стандартов (ЕСПД, UML).

## 3.Основные требования к курсовой работе

### 3.1. Тематика курсовой работы

Курсовая работа выполняется в инструментальных системах MS Visual Studio 2010/12/13/15. Каждая из этих систем предоставляет системную объектно-ориентированную среду на базе платформы Microsoft .NET Framework для разработки как настольных (клиентских) приложений, так и веб-приложений (в данной работе ограничимся классом настольных приложений). Например, предоставляется класс System::Console (класс Console в дальнейшем) для создания консольных приложений, класс Form для создания ОО приложений с графическим интерфейсом (приложений с программным интерфейсом Window Forms), классы Window и Application для создания приложений с интерфейсом WPF (Window Presentation Foundation).

Каждая из этих системных сред предоставляет разработчику ПО создать свое приложение, используя либо процедурно-ориентированный, либо объектно-ориентированный стиль программирования. Возможны и промежуточные варианты реализации приложения.

Например, при вычислении площади треугольника методом Монте-Карло диаграмма классов ОО реализации в среде WPF приложения может иметь вид, представленный на рис. 1. Эта диаграмма показывает взаимосвязь классов приложения. На рис. 2 показана более детальная диаграмма одного из классов приложения – класса треугольника.

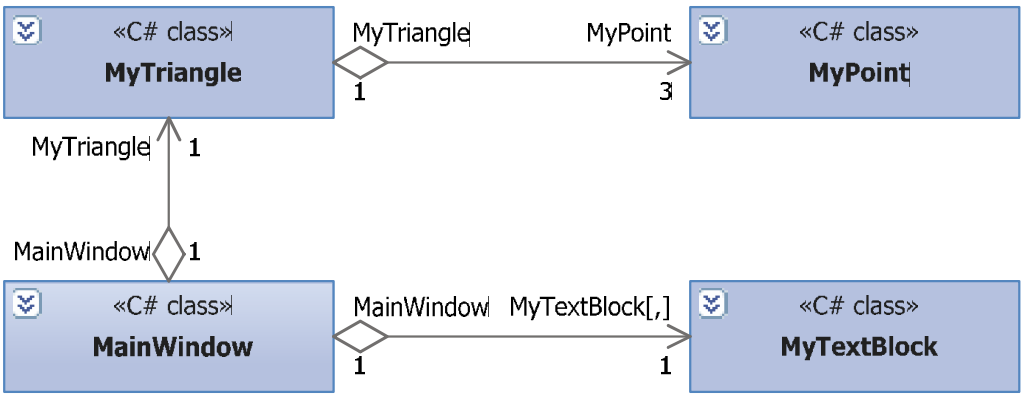


Рис. 1. Диаграмма классов настольного приложения WPF

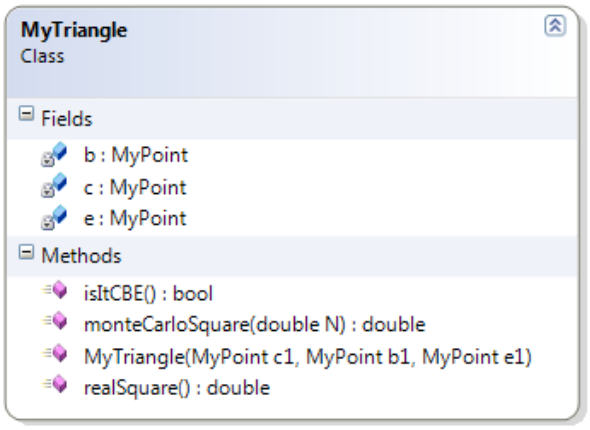


Рис. 2. Диаграмма класса MyTriangle настольного приложения

Отметим, что графический интерфейс приложения WPF описывается на декларативном языке XAML, используя ОО стиль программирования. Однако создание такого приложения в среде MS Visual Studio 2010/12/13/15 с использованием инструментальных панелей компонентов практически ничем не отличается от создания приложений Window Forms.

### 3.2. Исходные данные к курсовой работе

В результате выполнения курсовой работы необходимо создать и сравнить по вычислительной эффективности два приложения, решающие задачу приближенного вычисления площади геометрической фигуры методом Монте-Карло с использованием процедурного и объектно-ориентированного программирования соответственно.

В качестве геометрической фигуры в курсовой работе предлагается использовать одну из трех фигуру, представленных на рис. 3а, б,с. Здесь каждая линия отсекает одну четвертую часть окружности.

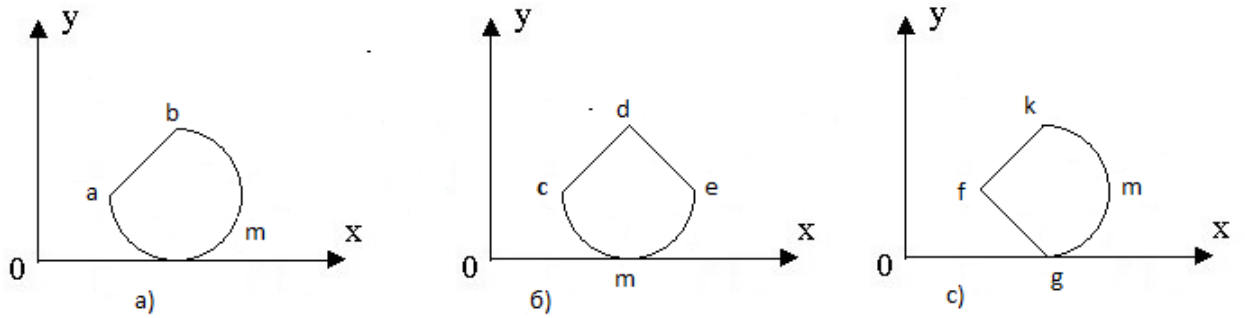


Рис. 3. Фигуры bma, demc и fkmg, вписанные в квадрат

Исходными данными для каждого приложения являются координаты угловых точек фигуры. Для однозначного определения конкретной фигуры достаточно ввести координаты всего двух точек. Координаты остальных точек вычисляются. Для вычисления площади фигуры необходимо вначале вычислить площадь прямоугольника, описывающего заданную фигуру, а затем $N$ раз сгенерировать по два случайных числа для координат x и y, определяющие точку внутри прямоугольника. Генерируемые случайным образом точки должны равномерно заполнять площадь прямоугольника. Для этого случайные числа должны иметь равномерное распределение (по ширине и высоте прямоугольника соответственно).

Для каждой точки выполняется проверка, попала ли точка внутрь заданной фигуры. Если из $N$ точек $M$ точек оказалось внутри фигуры, а площадь прямоугольника равна $S$, то площадь фигуры будет приближенно равна $S \cdot M / N$.

Поскольку площадь фигуры легко определяется по правилам геометрии, мы можем определить относительную погрешность приближенного вычисления этой площади методом Монте-Карло. Естественно, чем больше $N$, тем меньше погрешность такого вычисления.

Каждое приложение шесть раз повторяет эксперимент и вычисляет площадь фигуры методом Монте-Карло для $N = 10^3, 10^4, 10^5, 10^6, 10^7$ соответственно. В каждом эксперименте определяется относительная погрешность вычисления площади (в процентах) и его длительность (в миллисекундах).

По результатам экспериментов каждое приложение выдает на экран таблицу, показывающую зависимости значений относительной погрешности и длительности эксперимента от величины *N*.

Далее выполняется сравнительный анализ вычислительной эффективности процедурного и ОО приложений, а также профилирование работы приложений, нахождение узких мест. Определяются характеристики качества приложений. Затем используются рефакторинг, реинжиниринг и другие средства/методики улучшения характеристик качества приложений.

Курсовая работа по курсу "Объектно-ориентированное программирование" выполняется либо по реальному заданию, выданному преподавателем в рамках индивидуального задания по НИРС, либо по типовому заданию (табл. 1). По согласованию с преподавателем отдельные параметры типового задания могут быть изменены. Типовой вариант с измененными параметрами рассматривается как новый вариант.

В качестве индивидуальных заданий могут быть выбраны, например, как настольные, так и веб-приложения, широко использующие паттерны проектирования (не менее двух) и/или технологию LINQ.

В каждом варианте типового задания указываются:

- геометрическая фигура для вычисления площади методом Монте-Карло (см. рис. 3);
- программный интерфейс платформы Microsoft .NET Framework, в контексте которого будут создаваться и процедурное и объектно-ориентированное приложение (вариантами являются Console, WinForms и WPF);
- язык программирования – C++ или C#.

Таблица 1.
Варианты заданий для процедурных и ОО приложений

| Вариант | Фигура | Интерфейс приложения | Язык приложения |
|---|---|---|---|
| 1 | bma | WPF | C++ |
| 2 | bma | WPF | C# |
| 3 | bma | WinForms | C++ |
| 4 | bma | WinForms | C# |
| 5 | bma | Console | C++ |
| 6 | bma | Console | C# |
| 7 | demc | WPF | C++ |
| 8 | demc | WPF | C# |
| 9 | demc | WinForms | C++ |
| 10 | demc | WinForms | C# |
| 11 | demc | Console | C++ |
| 12 | demc | Console | C# |
| 13 | fkmg | WPF | C++ |
| 14 | fkmg | WPF | C# |
| 15 | fkmg | WinForms | C++ |
| 16 | fkmg | WinForms | C# |
| 17 | fkmg | Console | C++ |
| 18 | fkmg | Console | C# |

### 3.3. Задание на курсовую работу

При выполнении курсовой работы студент должен разработать входные и выходные документы и алгоритмы решения задачи приближенного вычисления площади фигуры методом Монте-Карло, реализовать алгоритмы, используя процедурный и объектно-ориентированный стили программирования, выполнить исследование вычислительной эффективности созданных приложений и представить результаты в виде таблиц, провести анализ и улучшить заданные характеристики качества приложений.

Задание на проектирование выдается студенту в течении первых двух недель пятого семестра. Задание оформляется на типовом бланке отдельно на каждый проект, включая индивидуальные задания. Образец бланка задания приведен в Приложении. На бланке типового задания указывается тема работы "Исследование вычислительной эффективности объектно-ориентированных приложений" и исходные данные, определенные по номеру варианта и заданные преподавателем. В течение первых четырех недель с момента выдачи задания исходные данные могут быть откорректированы по согласованию с руководителем работы. Индивидуальное задание должно быть согласовано с ведущим лектором курса "Объектно-ориентированное программирование".

### 3.4. Объем курсовой работы

Курсовая работа состоит из пояснительной записки и информационно-программных средств, реализующих задание на проектирование.

Пояснительная записка (ПЗ) оформляется согласно требованиям ЕСПД и должна содержать:

- титульный лист;
- бланк задания;
- введение;
- основное содержание;
- заключение;
- библиографический список;
- приложение, содержащее исходные тексты программ и изображения входных и выходных документов, подтверждающие выполнение основных этапов курсовой работы и отражающие ее отличия от других вариантов заданий.

Работающие варианты программ вместе с исходными текстами предоставляются преподавателю в электронном виде.

### 3.5. Защита курсовой работы

Выполненная и оформленная полностью курсовая работа предоставляется руководителю на проверку, который после проверки пояснительной записки и информационно-программного обеспечения подписывает ее к защите или возвращает студенту на доработку в зависимости от готовности работы.

Защищается курсовая работа перед комиссией из двух-трех преподавателей кафедры. Во время защиты студент докладывает об основных проблемах, возникших при разработке информационно-программного обеспечения и принятых им способах их решения, демонстрирует на компьютере создание приложений и использование веб-технологий в них, докладывает о полученных результатах и отвечает на вопросы членов комиссии.

Комиссия оценивает защиту курсовой работы как по качеству выполненной работы, так и по уровню знаний студента, проявленных им в процессе защиты.

В случае неудовлетворительной оценки студент получает новое задание и выполняет работу заново.

## 4. Методические указания к работе над курсовой работой

### 4.1. Основные этапы проектирования

Выполнение курсовой работы включает в себя следующие этапы:

Э1. Ознакомление с заданием, создание, настройка и освоение соответствующей заданию операционной среды (систем программирования и инструментальных систем разработки приложений) на компьютере.

Э2. Изучение алгоритма решения задачи приближенного вычисления площади геометрической фигуры методом Монте-Карло, ознакомление с возможностями генерирования случайных чисел и измерения интервалов времени в заданной системе программирования.

Э3. Разработка процедурного приложения (консольного или с графическим интерфейсом), реализующего алгоритм решения задачи приближенного вычисления площади фигуры методом Монте-Карло, проверка правильности работы алгоритма и выполнение исследования вычислительной эффективности алгоритма.

Э4. Разработка ОО приложения, реализующего алгоритм решения задачи приближенного вычисления площади четырехугольника методом Монте-Карло. При этом максимально следуют основным концепциям ООП. Выполняется исследования вычислительной эффективности ОО приложения.

Э5. Выполнение сравнительного анализа вычислительной эффективности процедурного и ОО приложений. Профилирование работы приложений, нахождение узких мест. Определение характеристик качества приложений. Использование рефакторинга, реинжиниринга и других средств/методик улучшения различных характеристик качества приложений.

Э6. Оформление пояснительной записки курсовой работы в соответствии с требованиями стандартов (ЕСПД, UML).

По согласованию с преподавателем некоторые из перечисленных этапов могут быть изменены.

### 4.2. Методические указания к выполнению отдельных этапов проектирования

Данная информация содержится в методических указаниях к лабораторным работам и в лекционном материале.

### 4.3. Содержание разделов пояснительной записки (текстовой части) к курсовой работе

В основное содержание ПЗ рекомендуется включить следующие разделы:

*Введение*. Дается краткое введение в технологии, которые будут использованы в данной курсовой работе.

1. *Постановка задачи.* Приводится задание на курсовую работу и формальное описание задачи, дается детальная постановка задачи проектирования с учетом исходных данных задания.
2. *Разработка технического задания.* Выполняется анализ функций создаваемого комплекса программ и разрабатывается диаграмма вариантов использования (Use Case–диаграмма) на языке UML. Приводятся требования к каждому из приложений комплекса.
3. *Анализ алгоритма решения*. Описывается алгоритм метода Монте-Карло и приводятся формулы для вычисления точной площади фигуры и относительной погрешности площади, полученной методом Монте-Карло. Описывается метод определения попадания точки внутрь фигуры, даются все математические выражения, которые используются в алгоритмах и будут реализованы в коде приложений.
4. *Процедурное приложение.* Дается информация о среде реализации настольного приложения, приводится описание программы (с использованием диаграмм на языке UML). Рассматривается реализация приложения и приводятся результаты работы.
5. *Объектно-ориентированное приложение.* Дается информация о среде реализации приложения, осуществляется выбор и приводится описание объектов и классов (с использованием диаграмм на языке UML). Рассматривается реализация приложения и приводится описание разработанного программного продукта в соотвествии с требованиями ЕСПД. Разрабатываются схемы программ (по ЕСПД) для основных методов приложения, а также приводятся основные фрагменты текстовых документов:
   - описание программы;
   - руководство программиста;
   - руководство системного программиста;
   - руководство оператора.
6. *Анализ вычислительной эффективности приложений.* Приводится информация о конфигурации компьютера и дается таблица, в которой приводятся результаты исследований.
7. *Улучшение характеристик качества приложений.* Приводятся результаты профилирования и определения характеристик качества приложений. Рассматриваются вопросы рефакторинга и реинжиниринга для улучшения заданных характеристик качества приложений. Приводятся полученные результаты.

Пояснительная записка выполняется на листах белой бумаги формата А4 (210х297 мм) машинописным или рукописным способом в соответствии с требованиями стандарта ЕСПД.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.701 - 90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения.

2. ГОСТ 19.101 - 77 ЕСПД. Виды программ и программных документов.

3. ГОСТ 19.106 - 78 ЕСПД. Требования к программным документам, выполненным печатным способом.

4. ГОСТ 19.401 - 78 ЕСПД. Текст программы. Требования к содержанию и оформлению.

5. ГОСТ 19.402 - 78 ЕСПД. Описание программы.

6. ГОСТ 19.404 - 79 ЕСПД. Пояснительная записка.Требования к содержанию и оформлению.

7. ГОСТ 28388 - 89 СОИ. Документы на магнитных носителях данных. Порядок выполнения и обращения.

8. ГОСТ 2.105 – 95 «Общие требования к текстовым документам»

9. ГОСТ 34.601-90 «Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания»

10. ГОСТ 34.201-89* . «Информационная технология. Комплекс стандартов на автоматизированные системы. Документация на АСУ. Виды, комплектность и обозначения документов при создании автоматизированных систем»

11. ГОСТ 51188-98. «Испытания программных средств на наличие компьютерных вирусов»

12. ГОСТ 19.503-79 Руководство системного программиста. Требования к содержанию и оформлению.

13. ГОСТ 19.504-79 Руководство программиста. Требования к содержанию и оформлению.

14. ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению.

15. Компьютерное оформление отчетных документов / Составитель Т.И. Матикашвили, ТулГУ, Тула, 2000. -36 с.

16. Орлов С. Технологии разработки программного обеспечения: Учебник. – СПб.: Питер, 2002. – 464 с.

17. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. Второе издание. Rational, Санта Клара, Калифорния. // Перевод с английского под редакцией И. Романовского и Ф. Андреева. – 1995.

18. Буч Г. и др. Язык UML. Руководство пользователя. 2-е изд. - СПб.: Питер, 2004. – 432 с. (Объектно-ориентированные технологии программирования)

19. Буч. Г., Якобсон А. Унифицированный процесс разработки программного обеспечения. - СПб.: Питер, 2002. – 496 с.

20. Буч Г. Язык UML : Руководство пользователя / Г.Буч, Д.Рамбо, И.Якобсон; пер.с англ.Мухин Н. — 2-е изд. — М. : ДМК Пресс:Академия Айти, 2007 .— 496с.

21. INTUIT.ru. Курс «Нотация и семантика языка UML». Автор А.В.Леоненков . Интернет университет информационных технологий, 2009.

22. INTUIT.ru. Курс «Введение в UML». Автор: А.В. Бабич. – Интернет университет информационных технологий, 2008.

23. INTUIT.ru . Курс «Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose» . Автор А.В.Леоненков .— Интернет университет информационных технологий, 2006 .

24. Розенберг Д., Скотт К. Применение объектного моделирования с использованием UML и анализ прецедентов. Учебное пособие / Д. Розенберг, К. Скотт. – М. : ДМК Пресс, 2007. – 160. (ЭБС IPRbooks)

25. Хабибуллин И.Ш. Самоучитель XML. – СПб.: БХВ-Петербург, 2003. – 336 с.

26. Троелсен Э. Язык программирования C# 2005 и платформа .NET 2.0. 3-е изд. – М.: ООО "И.Д. Вильямс", 2007. – 1168 с.

27. Троелсен, Эндрю. Язык программирования C# 2008 и платформа .NET 3.5 FRAMEWORK. - М. : ООО «И.Д. Вильямс», 2008. – 1344 с.

28. Макаров, А.В. Common Intermediate Language и системное программирование в Microsoft.NET : учебное пособие для вузов / А.В.Макаров,С.Ю.Скоробогатов,А.М.Чеповский .— М. : Интернет-Университет Информационных Технологий:Бином, 2006

29. Рихтер, Richter J. Программирование на платформе Microsoft .NET Framework 4.0 : пер.с англ. / Д.Рихтер .— 3-е изд. — М. : Питер, 2012 .— 928 с.. : ил. — (Мастер-класс)

30. Ч. Петцольд Microsoft Windows Presentation Foundation – М. : Русская Редакция; СПб. : Питер, 2008. – 944 с.

31. Ч. Петцольд. Программирование с использованием Microsoft Windows Forms. Мастер-класс. – М. : Русская Редакция; СПб. : Питер, 2006. – 432 с.

32. Пауэрс Л. Microsoft Visual Studio 2008 / Л. Пауэрс, М. Снелл: Пер. с англ. – СПб.: БХВ-Петербург, 2009. – 1200 с.

33. Макки А. Введение в .NET 4.0 и Visual Studio 2010 для профессионалов: Пер. с англ. – М.: ООО "И.Д. Вильямс", 2010. – 416 с.

34. Нэш Т. C# 2010: ускоренный курс для профессионалов: Пер. с англ. – М.: ООО "И.Д. Вильямс", 2010. – 592 с.

35. Подбельский, В.В. Язык Си+ : учеб.пособие для вузов / В.В.Подбельский .— 5-е изд. — М. : Финансы и статистика, 2007. — 560 с.

36. Павловская, Т.А. C/C++:Программирование на языке высокого уровня : учебник для вузов / Т.А.Павловская .— М.[и др.] : Питер, 2007 .— 461с.

37. Шилдт, Г. C+ : базовый курс / Г.Шилдт;пер.с англ.и ред.Н.М.Ручко .— 3-е изд. — М.[и др.] : Вильямс, 2007 (2005) .— 624с. : ил.

38. Г. Шилдт. Теория и практика C++ : Пер. с англ. – СПб.: BHV – Санкт-Петербург, 1999. – 416 с.

39. Уоткинз Д., Хаммонд М., Эйбрамз Б. Программирование на платформе .NET. – М.: Издательский дом "Вильямс", 2003. – 368 с.

40. Дубовцев А.В. Microsoft .NET в подлиннике. – СПб.: БХВ-Петербург, 2004. – 704 с.

41. Уоткинз Д., Хаммонд М., Эйбрамз Б. Программирование на платформе .NET. – М.: Издательский дом "Вильямс", 2003. – 368 с.

42. Цимбал А.А., Майоров А.Г., Козодаев М.А. Turbo C++:Пер. с англ.- М.: Джен Ай Лтд, 1993.- 512с.

43. С.Дьюхарст, К.Старк. Программирование на C++:Пер. с англ.- Киев: "ДиаСофт", 1993. - 272с.

44. Уотсон К., Нейгел К., Педерсен Я., Рид Д., Скиннер М. Visual C# 2010: полный курс.: Пер. с англ. - М.: ООО "И.Д. Вильяме", 2011. - 960 с.

45. Дейтел, Х.М. C# : пер.с англ. / Х.М.Дейтел [и др.] .— СПб. : БХВ-Петербург, 2006. - 1056с.

46. Жарков, В.А. Visual C# 2005 в учебе, науке и технике / В.А.Жарков .— М. : Жарков Пресс, 2007. - 818с.

47. Фаронов В. Программирование на языке C# : Учебный курс. – СПб., Питер, 2007. – 241 с.

48. Раттц-мл. Д.С. LINQ: язык интегрированных запросов в C# 2008 для профессионалов. - М.: ООО "И.Д. Вильямс", 2008. – 560 с.

49. Мартин Р., Мартин М. Принципы, паттерны и методики гибкой разработкина языке C#. – Пер. с англ. – СПб. : Символ-Плюс, 2011. – 768 с.

50. Фаулер М. Рефакторинг: улучшение существующего кода. - Пер. с англ. – СПб. : Символ-Плюс, 2003. – 432 с.

51. Петцольд Ч. Microsoft Windows Presentation Foundation. – М. : Издательство «Русская редакция»; СПб.: Питер, 2008. – 944 с.

52. Мак-Дональд Мэтью. WPF 4: Windows Presentation Foundation в .NET 4.0 с примерами на C# 2010 для профессионалов : Пер. с англ. – М.:ООО «И.Д. Вильямс», 2011. – 1024 с.