МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Воронежский государственный технический университет»

Кафедра автоматизированного оборудования
машиностроительного производства

АВТОМАТИЗИРОВАННЫЕ МЕТОДЫ ПРОЕКТИРОВАНИЯ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению практических работ
студентами направления подготовки
15.04.01 «Машиностроение»
(программа магистерской подготовки «Обеспечение качественно-точностных характеристик при изготовлении изделий
в автоматизированном машиностроительном производстве»)
очной и заочной форм обучения

Воронеж 2022

УДК 681.5:621.1.001.2(07)
ББК 32.966:30.606я7

Составитель С. Л. Новокщенов

Автоматизированные методы проектирования технологических процессов: методические указания к выполнению практических работ студентами направления подготовки 15.04.01 «Машиностроение» (программа магистерской подготовки «Обеспечение качественно-точностных характеристик при изготовлении изделий в автоматизированном машиностроительном производстве» очной и заочной форм обучения» / ФГБОУ ВО «Воронежский государственный технический университет»; сост. С. Л. Новокщенов. Воронеж: Изд-во ВГТУ, 2022. 38 с.

В методических указаниях изложены рекомендации к выполнению практических работ, приведены необходимые теоретические сведения, требования к оформлению отчетов, приведен библиографический список литературы. Выполнение практических работ направлено на получение навыков проектирования технологических и производственных процессов с применением современных ЭВМ и интегрированных сред разработки приложений.

Предназначены для студентов направления 15.04.01 «Машиностроение» (программа магистерской подготовки «Обеспечение качественно-точностных характеристик при изготовлении изделий в автоматизированном машиностроительном производстве» очной и заочной форм обучения».

Методические указания подготовлены в электронном виде и содержатся в файле АМПТП_ПР-577-2022.pdf.

Табл. 6. Ил. 12. Библиогр.: 5 назв.

УДК 681.5:621.1.001.2(07)
ББК 32.966:30.606я7

Рецензент: А. В. Демидов, канд. техн. наук, доцент кафедры автоматизированного оборудования машиностроительного производства ВГТУ

Издается по решению редакционно-издательского совета Воронежского государственного технического университета

ВВЕДЕНИЕ

Основной целью выполнения практических работ по дисциплине "Автоматизированные методы проектирования технологических процессов" является ознакомление с возможностями решения задач проектирования технологических и производственных процессов с применением современных ЭВМ и интегрированных сред разработки приложений.

Тематика и содержание работ составлены с учётом материальной базы лаборатории кафедры и с учётом возможности самостоятельного выполнения с применением ресурсов сети Internet.

Постоянный технический прогресс, возрастающая конкуренция среди производителей, увеличение номенклатуры выпускаемой продукции во всех типах производств и высокая скорость ее обновления заставляют искать резервы сокращения времени на подготовку производства новых изделий, что обеспечивается конструкторскими, организационными и технологическими мероприятиями. В настоящее время одним из перспективных направлений обеспечения конкурентоспособности предприятия является повышение эффективности технологической подготовки производства (ТПП) выпускаемых изделий. Целью ТПП является оптимальное по срокам и ресурсам обеспечение технологической готовности производства к изготовлению изделий в соответствии с требованиями заказчика или рынка данного класса изделий.

Необходимость повышения эффективности ТПП объясняется еще и тем, что проектирование технологической документации в большинстве случаев значительно более трудоемко, чем разработка конструкторской. Ощутимое повышение эффективности ТПП по сравнению с существующим уровнем возможно только при выполнении следующих условий:

• наличия единого информационного пространства для специалистов конструкторской и технологической служб предприятия;

• повышения скорости разработки и обоснованности планов ТПП;

• непрерывного контроля их выполнения.

С появлением широкодоступных персональных компьютеров и рабочих станций стали возможными: обеспечение каждого пользователя индивидуальным автоматизированным рабочим местом; организация вычислительных сетей; работа в интерактивном графическом режиме; электронный обмен данными; организация единых централизованных и распределенных баз данных; решение задач, требующих больших вычислительных ресурсов. Благодаря этому появилась возможность автоматизировать проектирование любых ТП (программы класса CAM (Computer Aided Manufacturing)). Открывшиеся инструментальные и программные возможности активизировали работы по созданию АСТПП (программы класса класса PDM (Product Data Management)).

Более чем за 40 лет были сформулированы и проверены на практике следующие основополагающие принципы построения АСТПП.

• Системного единства. Элементы АСТПП должны разрабатываться как части единого целого, их функционирование подчинено общей цели. Кроме того, должна обеспечиваться интеграция АСТПП с автоматизированной системой управления производством (АСУП).

• Декомпозиции. Разделение АСТПП на составляющие (подсистемы) должно быть выполнено по наиболее слабым организационным и информационным связям. Правильная декомпозиция уменьшает сложность системы и облегчает условия ее эксплуатации.

• Модульности. Все компоненты АСТПП должны представлять собой логически независимые модули, которые

могут использоваться как в автономном, так и в комплексном режиме.

• Совместимости. Все компоненты АСТПП должны обеспечивать возможность их совместного функционирования. Это требует их организационной, информационной и программной совместимости.

• Открытости. На этапе создания АСТПП невозможно предусмотреть все нюансы и перспективы дальнейшего развития производства, поэтому она должна быть открыта для модернизации и включения в нее новых решений.

• Стандартизации. В АСТПП должно быть использовано максимальное число унифицированных, типовых и стандартных решений. Это уменьшает затраты на ее создание, повышает надежность функционирования.

• Эргономичности. Поскольку АСТПП является человеко-машинной системой, следует предусматривать удобство работы се пользователей (правильное разделение функций, удобство и простоту интерфейсов, учет психологических факторов и др.).

• Ориентации на новые достижения. При создании АСТПП должны использоваться последние научно-технические достижения в области методов ее построения, методов и средств ТПП, организации производства. Из изложенного, однако, не должен следовать вывод, что весь процесс подготовки производства можно полностью.

Одной из важнейших составных частей ТПП, как уже отмечалось ранее, является проектирование собственно ТП. По возможности необходимо не заниматься разработкой собственных алгоритмов и программ, а стремиться к поиску уже готовых, которые, с одной стороны, отвечают необходимым функциональным требованиям, с другой – уже доказали свою надежность и качество при использовании для других аналогичных изделий или на других предприятиях.

Раздел 1. Рекомендации по изучению дисциплины и выполнению практических работ

ОРГАНИЗАЦИЯ ЗАНЯТИЙ

Занятия в лаборатории проводятся под руководством преподавателя. Для проведения практических занятий группа делится на подгруппы (по 10 - 12 человек), постоянный состав которых сохраняется до окончания всего цикла практических работ. Практические работы выполняется студентами самостоятельно. По результатам выполненных работ оформляется отчет. По окончании цикла практических работ каждый студент должен сдать зачёт. При сдаче зачёта студент обязан:

1. Знать целевое назначение работы и уметь объяснить порядок и технику её выполнения.

2. Знать устройство, приемы управления и настройку оборудования, приборов и программных средств, применяемых в работе.

3. Понимать физический и практический смысл полученных результатов.

4. Предъявить отчёт с записями со всеми необходимыми расчётами, эскизами, графиками и выводами по каждой выполненной работе.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТ

Перед началом занятий студенты знакомятся с содержанием цикла практических работ, организацией и режимом занятий, правилами техники безопасности. Распределение обязанностей внутри подгруппы производится студентами с соблюдением принципа равного участия в работе каждого студента.

Студенты должны:

1. Изучить самостоятельно методику выполнения работы и ознакомиться с организацией рабочего места.

2. Ознакомиться под руководством преподавателя или лаборанта с устройством лабораторного оборудования и его управлением.

3. Категорически запрещается самостоятельный пуск оборудования и пользование без ведома преподавателя или лаборанта.

4. Изучить правила техники безопасности.

5. Произвести под руководством преподавателя или лаборанта настройку оборудования и приборов.

6. Выполнить самостоятельно необходимые учебные задания в соответствии с методикой. Результаты занести в рабочую тетрадь.

7. После окончания работы рабочее место сдать лаборанту.

8. Провести анализ полученных результатов и сделать выводы по работе. Оформить и сдать преподавателю отчет.

ТРЕБОВАНИЯ К ОТЧЕТУ

Отчет по работе оформляется на бумаге стандартного формата (формат А4). Отчет брошюруется в общую тетрадь. Отчет представляется в печатном виде. Коллективное составление и сдача отчетов не допускается.

Отчет по практической работе должен быть выполнен в текстовом редакторе Microsoft Word 2010 или выше и содержать: титульный лист, название темы работы, цели работы, перечень технических и программных средств, необходимых для выполнения практической работы; краткое описание исследуемого вопроса; алгоритм программы; исходные данные варианта; распечатку полученных в ходе расчета значений; выводы, содержащие анализ проведенной работы.

В выводах дается краткое объяснение сущности полученных результатов. Выводы должны быть краткими и отвечать на вопросы, поставленные в практической работе.

ТЕХНИКА БЕЗОПАСНОСТИ ПРИ РАБОТЕ СТУДЕНТОВ В ЛАБОРАТОРИИ

Для того чтобы уберечь себя и товарищей от несчастного случая, а государственное имущество от аварии, необходимо хорошо знать и полностью выполнять правила внутреннего распорядка, техники безопасности и пожарной безопасности. К практическим работам допускаются студенты, которые ознакомились с общими конкретными требованиями техники безопасности и прошли соответствующий инструктаж.

Проведение инструктажа и проверка знаний правил техники безопасности должны быть зарегистрированы соответствующими записями в лабораторном журнале. Конкретные требования техники безопасности при проведении той или иной работы изложены в описании к практическим работам.

Раздел 2. Практические занятия

Практическая работа №1
Разработка алгоритмов программных модулей автоматизации проектирования исследуемого технологического процесса

(3 часа)

Цель работы: ознакомиться с возможностями программирования с применением современных интегрированных рабочих сред, автоматизировать решение задачи с применением современных технологий.

Технические средства и программное обеспечение:

1. IBM-PC или совместимый компьютер;
2. Операционная система Microsoft Windows;

3. Пакет офисных программ Microsoft Office;
4. Microsoft Visual Studio Community.

Теоретические сведения:

Для успешного взаимодействия с техническими средствами и рационального их использования проектировщик обязан овладеть определенной системой знаний, приобрести соответствующие навыки, умение и опыт. Это необходимо, прежде всего в профессиональной сфере деятельности - для автоматизации инженерных расчетных методик посредством программирования.

Программирование является тем видом деятельности, в котором органически сочетаются методы анализа и синтеза. Поэтому для программиста крайне важны способность к обобщению, конструктивность мышления, точное восприятие обстановки, осознание ограничений со стороны технических и программных средств, он должен соотносить время работы программы и объемы занимаемой памяти.

В настоящее время для анализа и управления данными при разработке деталей машин или технологическом проектировании можно использовать специализированные программные модули или созданные самостоятельно при помощи современных языков визуального программирования, таких, как Microsoft Visual Basic.NET применительно к среде Microsoft Visual Studio. VisualBasic.NET (VB.NET) — объектно-ориентированный язык программирования, который можно рассматривать как очередной виток эволюции Visual Basic (VB), реализованный на платформе Microsoft .NET.

Рассмотрим пример создания первого приложения на языке Visual Basic.NET. Для облегчения написания программного кода обычно используют специальные среды разработки, которые предоставляют многообразие различных возможностей по созданию программ. Наиболее часто применяемой при программировании средой разработки на VB.NET является Microsoft Visual Studio. Основы програм-

мирования в настоящем курсе лабораторных работы будем рассматривать применительно к языку Microsoft Visual Basic.

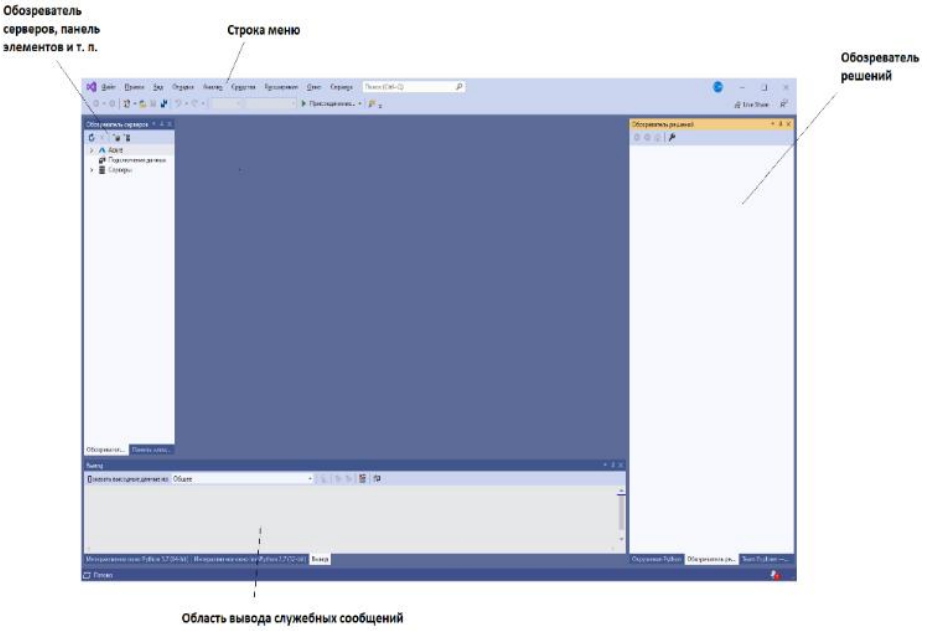


Рис. 1. Интерфейс IDE Microsoft Visual Studio

Самая простая программа состоит из следующих частей.

Здесь следует сделать замечания по особенностям синтаксиса VB.NET:

- концы кода **не помечаются** специальными символами, например, точкой с запятой (;);

- строки комментариев начинаются с символа «апостроф» (`);

- для выделения блоков кода **не используются** фигурные скобки { и };

- VB.NET **не чувствителен** к регистру.

VB.NET - инструмент для быстрого создания Windows-приложений. После смены концепции язык обрел:

1. поддержку объектно-ориентированного программирования с конструкторами, деструкторами и наследованием;
2. свободную многопоточность;
3. компиляцию в байт-код, исполняемую при помощи CLR;
4. структурную обработку исключений.

Microsoft Visual Basic.NET работает со следующими структурами:

- Классы – ***class***;
- Переменные – ***dim***. Переменные могут быть следующих основных типов:
- ***single***;
- ***integer***;
- ***decimal***;
- Константы – ***const***;
- Процедуры – ***sub***;
- Функции – ***function***.

Базовые команды, осуществляющие управление программными данными, в целом аналогичны большинству языков программирования, но в отличие от них имеют более простой синтаксис. Код программы пишется в текстовом редакторе, войти в редактор кода можно несколькими способами. Программы для ЭВМ создаются на основе заранее разработанного ***алгоритма***.

```
Public Class Form1 `Объявление имени модуля

    Private Sub Form1_Load(sender As Object, e As EventArgs)
 Handles MyBase.Load `Объявления имени процедуры начальной
 загрузки

    End Sub `Конец процедуры

End Class
```

Рис. 2. Структура программы

Наиболее сложными являются предварительная разработка алгоритма технологического проектирования и состав-

ление программы работы компьютера. Алгоритм и программа могут разрабатываться для специального и типового вариантов проектирования. В последнем случае по единой программе решаются задачи, сходные по структуре и последовательности выполнения этапов (проектирование технологии изготовления типовых деталей разных размеров). В компьютер каждый раз вводятся исходные данные и ограничивающие условия. Основные этапы автоматизированного проектирования технологии на компьютере приведены на рис. 3.

Рис. 3. Основные этапы разработки технологического процесса с помощью ЭВМ

Алгоритм - предписание, однозначно задающее процесс преобразования исходной информации в виде последовательности элементарных дискретных шагов, приводящих за конечное число их применений к результату.

Если вычислительный процесс заканчивается получением результатов, то говорят, что соответствующий алгоритм применим к рассматриваемой совокупности исходных данных. В противном случае говорят, что алгоритм неприменим к совокупности исходных данных. Любой применимый алгоритм обладает следующими основными свойствами:

1) дискретностью;

2) определенностью;
3) результативностью;
4) массовостью.

Дискретность – последовательное выполнение простых или ранее определённых (подпрограммы) шагов. Преобразование исходных данных в результат осуществляется дискретно во времени.

Определенность состоит в совпадении получаемых результатов независимо от пользователя и применяемых технических средств (однозначность толкования инструкций).

Результативность означает возможность получения результата после выполнения конечного количества операций.

Массовость заключается в возможности применения алгоритма к целому классу однотипных задач, различающихся конкретными значениями исходных данных (разработка в общем виде).

Для задания алгоритма необходимо описать следующие его элементы:

1) набор объектов, составляющих совокупность возможных исходных данных, промежуточных и конечных результатов;
2) правило начала;
3) правило непосредственной переработки информации (описание последовательности действий);
4) правило окончания;

К основным способам описания алгоритмов можно отнести следующие:

1) словесно-формульный (на естественном языке);
2) структурный или блок-схемный;
3) с использованием специальных алгоритмических языков;
4) с помощью граф-схем (граф - совокупность точек и линий, в которой каждая линия соединяет две точки. Точки называются вершинами, линии - рёбрами);
5) с помощью сетей Петри.
6) правило извлечения результатов.

Блок-схемы

При блок-схемном описании алгоритм изображается геометрическими фигурами (блоками), связанными по управлению линиями (направлениями потока) со стрелками. В блоках записывается последовательность действий.

Данный способ по сравнению с другими способами записи алгоритма имеет ряд преимуществ. Он наиболее нагляден: каждая операция вычислительного процесса изображается отдельной геометрической фигурой. Кроме того, графическое изображение алгоритма наглядно показывает разветвления путей решения задачи в зависимости от различных условий, повторение отдельных этапов вычислительного процесса и другие детали. Оформление программ должно соответствовать определенным требованиям. В настоящее время действует единая система программной документации (ЕСПД), которая устанавливает правила разработки, оформления программ и программной документации.

В ЕСПД определены и правила оформления блок-схем алгоритмов (ГОСТ 10.002-80 ЕСПД, ГОСТ 10.003-80 ЕСПД). Операции обработки данных и носители информации изображаются на схеме соответствующими блоками. Большая часть блоков по построению условно вписана в прямоугольник со сторонами a и b. Минимальное значение, a равно 10 мм, увеличение, a производится на число, кратное 5 мм. Размер b=1,5 мм. Для отдельных блоков допускается соотношение между a и b, равное 1:2. В пределах одной схемы рекомендуется изображать блоки одинаковых размеров. Все блоки нумеруются. Виды и назначение основных блоков приведены в таблице. Линии, соединяющие блоки и указывающие последовательность связей между ними, должны проводиться параллельно линиям рамки. Стрелка в конце линии может не ставиться, если линия аправлена слева направо или сверху вниз. В блок может входить несколько линий, то есть блок может являться преемником любого числа блоков.

Из блока (кроме логического) может выходить только одна линия. Логический блок может иметь в качестве продолжения одни из двух блоков, и из него выходят две линии. Если на схеме имеет место слияние линий, то место пересечения выделяется точкой. В случае, когда одна линия подходит к другой и слияние их явно выражено, точку можно не ставить. Схему алгоритма следует выполнять как единое целое, однако в случае необходимости допускается обрывать линии, соединяющие блоки. Если при обрыве линии продолжение схемы находится на этом же листе, то на одном и другом конце линии изображается специальный символ соединитель — окружность диаметром 0,5 мм. Внутри парных окружностей указывается один и тот же идентификатор. В качестве идентификатора, как правило, используется порядковый номер блока, к которому направлена соединительная линия. Если схема занимает более одного листа, то в случае разрыва линии вместо окружности используется межстраничный соединитель. Внутри каждого соединителя указывается адрес — откуда и куда направлена соединительная линия. Адрес записывается в две строки: в первой указывается номер листа, во второй — порядковый номер блока.

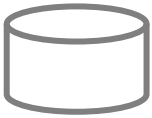
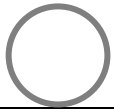
Блок-схема должна содержать все разветвления, циклы и обращения к подпрограммам, содержащиеся в программе.

Таблица 1

Описание элементов блок-схем

Элемент	Графическое обозначение	Описание
Процесс		Выполнение операции или группы операции, в результате которых изменяется значение, форма представления или расположение данных.

Продолжение табл.1

1	2	3
Ввод-вывод		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
Решение		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условии.
Предопределенный процесс		Использование ранее созданных и отдельно написанных программ (подпрограмм)
Документ		Вывод данных на бумажный носитель.
Магнитный диск		Ввод-вывод данных, носителем которых служит магнитный диск.
Вывод на монитор		Вывод на монитор
Пуск-останов		Начало, конец, прерывание процесса обработки данных.
Соединитель		Указание связи между прерванными линиями, соединяющими блоки.

Окончание табл.1

1	2	3
Межстраничный соединитель		Указание связи между прерванными линиями, соединяющими блоки, расположенные на разных листах.

Более сложным является алгоритм расчета траектории движения инструмента или припусков и промежуточных размеров при обработке. В качестве исходной информации используют следующие данные: чертеж детали с техническими требованиями; метод получения, точность и качество поверхности заготовок; установочные базы; тип приспособления; технологические маршруты обработки элементарных поверхностей; вид и место термической обработки в структуре ТП обработки элементарной поверхности. Укрупненная блок-схема алгоритма расчета припусков и промежуточных размеров наружной поверхности детали типа тела вращения приведена на рис. 4.

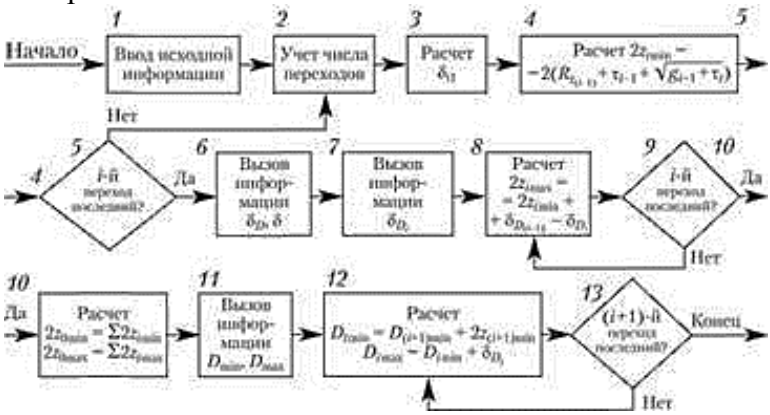


Рис.4. Блок-схема алгоритма расчета припусков на обработку и промежуточных размеров

Первая простая консольная программа.

```
Module Module1

    Sub Main()
        Console.WriteLine("Hello") –
выводит в терминал строку с текстом
        Console.ReadKey(True) – ожидает
нажатия любой клавиши
    End Sub

End Module
```

Рис. 5. Первое консольная программа на языке программирования Microsoft Visual Basic.NET

Рис. 6. Алгоритм первой консольной программы

Кроме языка VisualBasic.NET для реализацией модулей автоматизации решения технологических задач можно применять язык программирования Python.

Python представляет популярный высокоуровневый язык программирования, который предназначен для создания приложений различных типов. Это и веб-приложения, и игры, и настольные программы, и работа с базами данных. Довольно большое распространение питон получил в области машинного обучения и исследований искусственного интеллекта. Впервые язык Python был анонсирован в 1991 году голландским разработчиком Гвидо Ван Россумом.

Python - это интерпретируемый язык программирования, это означает, что как разработчик вы пишете файлы Python .py в текстовом редакторе, а затем помещаете эти файлы в интерпретатор Python для выполнения.

Структура программы на языке Python выглядит следующим образом (рис. 7).

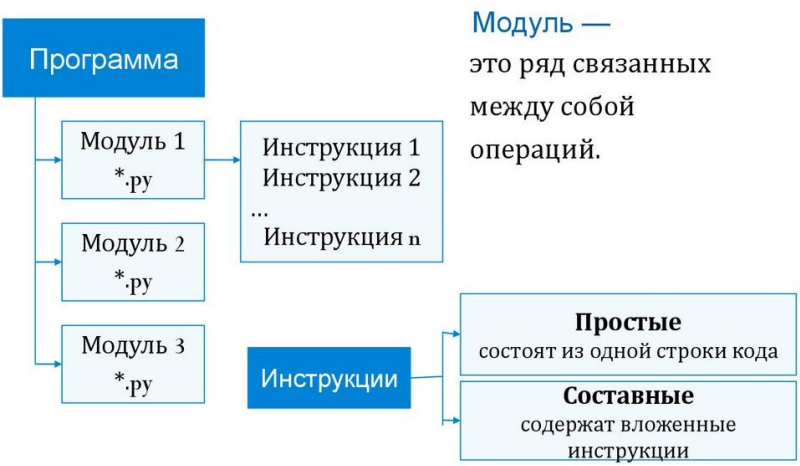


Рис. 7. Структура программы на языке Python

Опыт использования ЭВМ для проектирования ТП обработки резанием показал, что трудоемкость снижается в 10–15 раз, себестоимость в 2–4 раза по сравнению с обычными методами проектирования; оптимизация станочных операций повышает производительность обработки на 20–30% и снижает ее себестоимость на 10–15%.

Выполнение работы:

1) Предоставить преподавателю тематику работы, основные формулы, исходные данные и получаемые результаты;

2) С помощью средств табличного процессора Microsoft Excel разработать алгоритмы автоматизации расчетных методик;

3) С помощью средств IDE Microsoft Visual Studio Community и средств языка программирования Microsoft Visual Basic разработать приложение для операционной системы Microsoft Windows, получающее и анализирующее полученные результаты при решении задачи на ЭВМ.

Практическая работа №2
Разработка программных модулей автоматизации расчетных методик исследуемого технологического процесса

(3 часа)

Цель работы: ознакомиться с вопросами нормирования изучаемого технологического процесса, разработать алгоритм решения задачи технологического нормирования на ЭВМ.

Технические средства и программное обеспечение:

1. IBM-PC или совместимый компьютер;
2. Операционная система Microsoft Windows;

3. Пакет офисных программ Microsoft Office;
4. Microsoft Visual Studio Community.

Теоретические сведения:

На основе созданного алгоритма достаточно просто разработать приложение, которое позволит осуществлять многовариантный многокритериальный анализ с учетом изменения внешних условий или управляющих параметров

В общем случае используем среду NET.Framework для создания приложения, отвечающего всем требованиям операционных систем Microsoft Windows. Далее рассмотрим свойства формы такого приложения, устанавливаемые по умолчанию. В случае применения IDE Community 2019 или выше, вначале, при создании проекта, необходимо выбрать следующее (рис. 8).

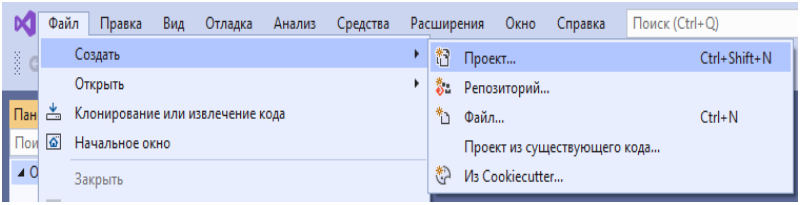


Рис. 8. Создание нового проекта

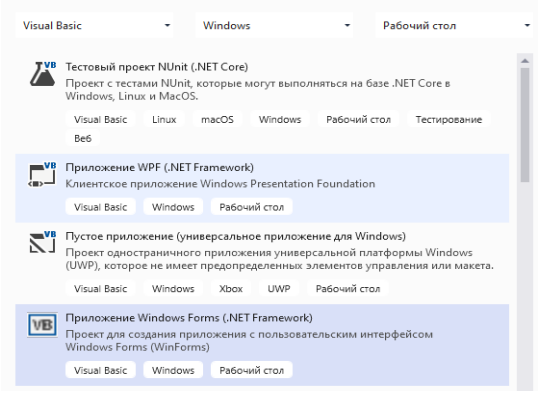


Рис. 9. Выбор среды реализации

Далее откроются инструменты (слева), редактор формы (по центру) и окно свойств элементов (справа) формы (рис. 10).

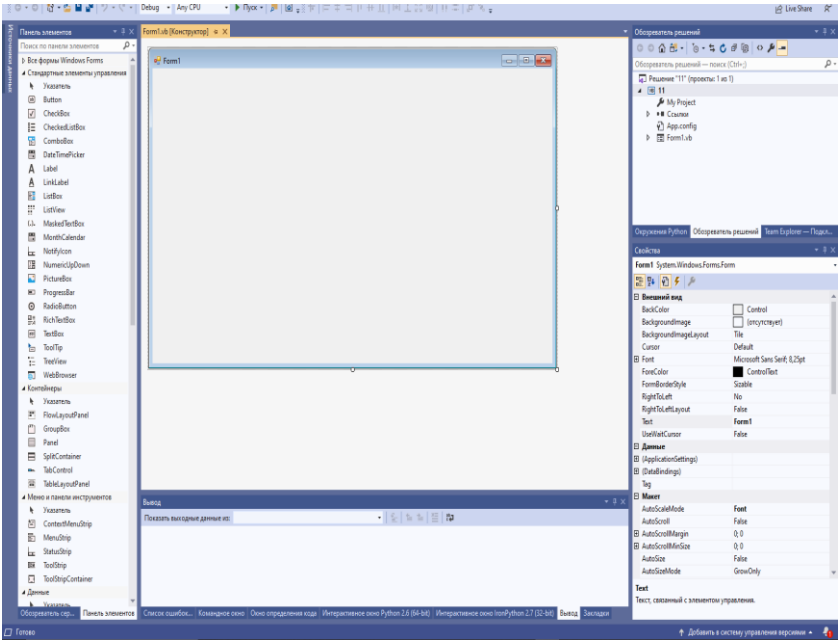


Рис. 10. Основные инструменты и возможности

Поскольку в данном случае речь идет об объектно-ориентированном программировании, то следует помнить, что каждому объекту соответствует определенный набор свойств, которые настраивают его внешний вид и определяют реакции на определенные действия пользователя.

Свойства формы будут доступны после двойного щелчка левой кнопкой мыши по ней. Для нашего случая рекомендуется предварительно настроить размеры формы и установить следующие значения параметров:

1) Запрет разворачивания окна во весь экран (рис. 11).

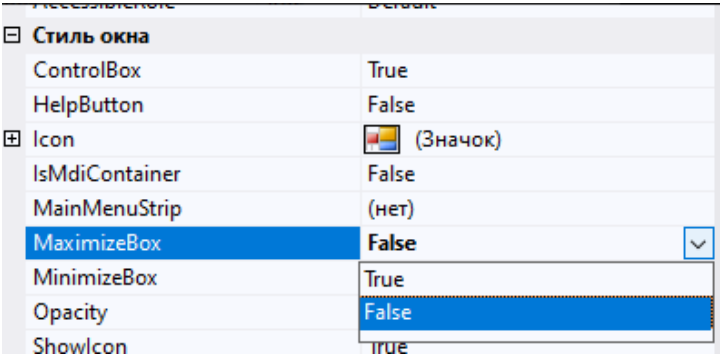


Рис. 11

2) Запрет изменения границ формы (рис. 12).

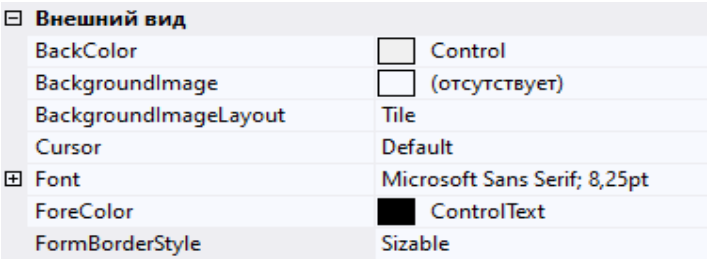


Рис. 12

3) Запрет изменения границ формы (рис. 13)

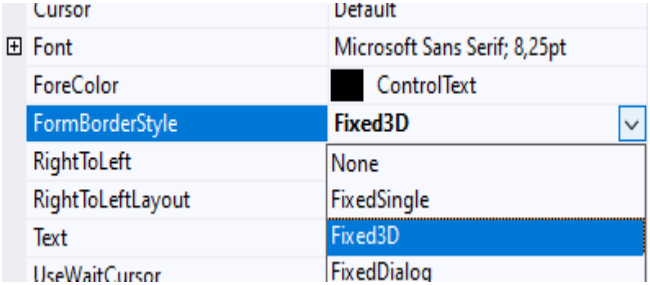


Рис. 13

4) Название программы

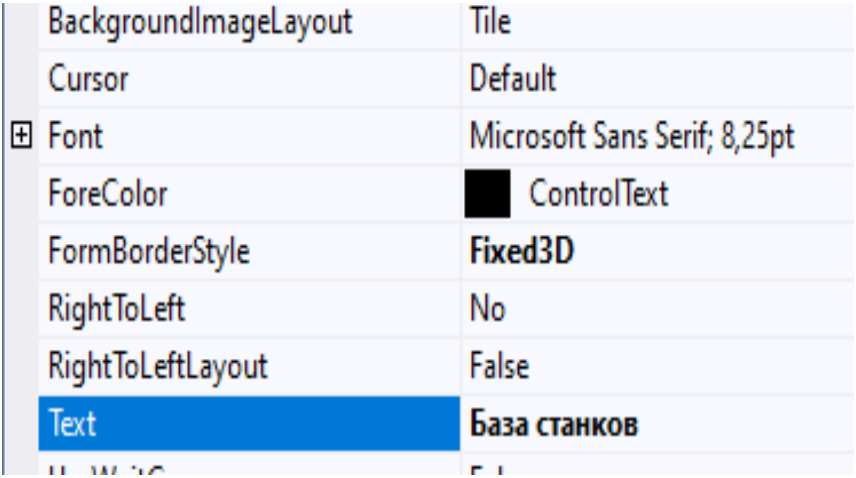


Рис. 14

Реализация модулей автоматизации решения конструкторско-технологических задач в языке программирования Python

В системах Windows и Macintosh интерпретатор можно запустить как приложение (либо из меню Start, либо двойным щелчком на пиктограмме интерпретатора). После запуска интерпретатора появляется подсказка, в которой можно начать отладку операторов программы в простом цикле чтения/выполнения. Например, в приведенном ниже выводе интерпретатор отображает сообщение об авторских правах и предоставляет пользователю подсказку >>>, в которой пользователь набирает знакомую команду "Hello World":

Исходные файлы Python имеют расширение *.py. Символ # в предыдущем примере обозначает комментарий, который продолжается до конца строки.

Основные инструкции языка Python приведены в табл. 2.

Таблица 2

Инструкции языка Python

Инструкция	Роль	Пример
1	2	3
Присваивание		
Вызовы	Создание ссылок	f.write('Пролог\n')
print	Вывод на консоль	print 'Знание – сила'
if/elif/else	Операция выбора	if 'python' in text: print text
for/else	Обработка последовательности в цикле	for x in thelist: print x
while/else	Цикл общего назначения	while x>y: y+=1
pass	Пустая инструкция	if a: pass
break, continue П	Переходы в теле цикла	while 1: if not in line: break
try/except/ finally	Обработка исключений	try: action() except: print 'action error'
raise	Возбуждение исключений	raise endSearch, location
import, from	Доступ к модулям	import sys from sys import stdin
def, return, yield	Создание функции	def f(a,b,c=1,*d): return a+b+c+d[0] def gen(n): for i in n, yield i*2
class	Описание класса	class subclass(Superclass): staticData=[]
global	Пространство имен	def function(): global x,y x='new
del	Удаление ссылок del	data[k]

Окончание табл. 2

1	2	3
exec	Запуск фрагментов программного кода	exec 'import '+modName exec code in gdict, ldict
assert	Отладочные проверки	assert x > y
with/as	Менеджеры контекста	with open('data') as myfile: process(myfile)

Python – это язык с динамическим контролем типа, в котором имена во время выполнения программы могут представлять значения различных типов. И действительно, имена, используемые в программе, – это только метки для различных величин и объектов. Оператор присваивания просто создает связь между именем и значением. В этом состоит одно из отличий данного языка, например, от C, в котором имена представлены объектами с постоянным размером и размещением в памяти, где находятся результаты.

Все данные в Python представлены объектами. Имена являются лишь ссылками на эти объекты и не несут нагрузки по декларации типа. Значения встроенных типов имеют специальную поддержку в синтаксисе языка: можно записать литерал строки, числа, списка, кортежа, словаря (и их разновидностей). Синтаксическую же поддержку операций над встроенными типами можно легко сделать доступной и для объектов определяемых пользователями классов. Следует также отметить, что объекты могут быть неизменчивыми и изменчивыми. Например, строки в Python являются неизменчивыми, поэтому операции над строками создают новые строки.

Карта встроенных типов (с именами функций для приведения к нужному типу и именами классов для наследования от этих типов):

1. Специальные типы: None, NotImplemented и Ellipsis;
2. Числа:
1) целые:
- обычное целое int;
- целое произвольной точности long;
- логическое bool;
2) число с плавающей точкой float;
3) комплексное число complex.
3. Последовательности:
1) неизменчивые:
- строка str;
- Unicode-строка Unicod;
- кортеж tuple;
2) изменчивые:
- список list;
- отображения;
- словарь dict.
4. Объекты, которые можно вызвать:
1) функции (пользовательские и встроенные);
2) функции-генераторы;
3) методы (пользовательские и встроенные);
4) классы (новые и «классические»);
5) экземпляры классов (если имеют метод _call_).
5. Модули.
6. Файлы file.
7. Вспомогательные типы buffer, slice.

Узнать тип любого объекта можно с помощью встроенной функции type().

ВЫРАЖЕНИЯ

В современных языках программирования принято производить большую часть обработки данных в выражениях. Синтаксис выражений у многих языков программирования примерно одинаков. Синтаксис выражений Python не удивит программиста чем-то новым. (Разве что цепочечные сравне-

ния могут приятно порадовать.) Приоритет операций показан в таблице (в порядке уменьшения). Для унарных операций x обозначает операнд. Ассоциативность операций в Python – слева направо

Таблица 3

Операция	Название
1	2
lambda	Лямбда-выражение
or	Логическое ИЛИ
and	Логическое И
not x	Логическое НЕ
in, not in	Проверка принадлежности
Is, is not	Проверка идентичности
<, <=, >, >=, !=, ==	Сравнения
|	Побитовое ИЛИ
^	Побитовое исключающее ИЛИ
&	Побитовое И
<<. >>	Побитовые сдвиги
+, -	Сложение и вычитание
*, /, %	Умножение, деление, остаток
+x, -x	Определение и смена знака
**	Возведение в степень
x атрибут	Ссылка на атрибут
x[индекс]	Взятие элемента по индексу
x[от: до]	Выделение среза (от и до)
f(аргумент)	Вызов функции
(…)	Скобки или кортеж
[…]	Список или списковое включение
{кл:зн,…}	Словарь пар ключ-значение
‘выражения’	Преобразование к строке (repr)

Порядок вычислений операндов определяется правилами:

1. Операнд слева вычисляется раньше операнда справа во всех бинарных операциях, кроме возведения в степень.

2. Цепочка сравнений вида a < b < c < … < y < z фактически равносильна данному выражению: (a < b) and (b < c) and ... and (y < z).

3. Перед фактическим выполнением операции вычисляются нужные для нее операнды. В большинстве бинарных операций предварительно вычисляются оба операнда (сначала левый), но операции or и and, а также цепочки сравнений вычисляют такое количество операндов, которого достаточно для получения результата. В невычисленной части выражения в таком случае могут даже быть неопределенные имена. Это важно учитывать, если используются функции с побочными эффектами.

4. Аргументы функций, выражения для списков, кортежей, словарей и т. п. вычисляются слева направо, в порядке следования в выражении. В случае неясности приоритетов желательно применять скобки, несмотря на то что одни и те же символы могут использоваться для разных операций, приоритеты которых не меняются. Так, «%» имеет тот же приоритет, что и «*», а потому в следующем примере скобки просто необходимы, чтобы операция умножения произошла перед операцией форматирования.

Повторный импорт модуля происходит гораздо быстрее, так как модули кэшируются интерпретатором. Загруженный модуль можно загрузить еще раз (например, если файл с текстом модуля изменился на диске) с помощью функции reload(). Однако в этом случае все объекты, являющиеся экземплярами классов из старого варианта модуля, не изменят своего поведения.

ФУНКЦИИ

Для создания функции применяется оператор def, как показано в следующем примере:

```
def get_temp_comfort(temp):
#создание словаря с выбором по значению
   return {
           temp <= -20: 'холодно',
      -20 <= temp <= 0:   'прохладно',
        0 <= temp <= 15:  'зябко',
       15 <= temp <=  25: 'тепло',
       25 <= temp:        'жарко'
   }[True]
```

Для того чтобы вызвать функцию, нужно просто указать имя функции, за которым следуют ее параметры, заключенные в круглые скобки. Для возврата из функции нескольких значений может применяться кортеж, как показано ниже:

```
temp=float (input("Введите значение температуры окружающего воздуха, T = ")) #Ввод значения температуры
if (temp >= -20 and temp <= 25): #Проверка вхождения значения в допустимый интервал
   print("В производственных помещениях: ",get_temp_comfort(temp))
else:
   print("Значение температуры выходит за допустимые пределы!")
```

ВСТРОЕННЫЕ ФУНКЦИИ

В среде Python без дополнительных операций импорта доступно более сотни встроенных объектов, в основном функций и исключений. Для удобства функции можно условно разделить на следующие категории:

Таблица 4

Операция	Название
1	2
Функции преобразования типов и классы	coerce, str, repr, int, list,tuple,long,float, complex, dict, super, file, bool, object
Числовые и строковые функции	abs, divmod, ord, pow, len, chr, unichr,hex,oct,cmp,round, Unicode
Функции обработки данных	apply, map, filter, reduce, zip, range, xrange, max, min, iter, enumerate, sum
Функции обработки данных	hash, id, callable, issubclass, isinstance, type
Функции для доступа к внутренним структурам	locals, globals, vars, intern, dir
Функции компиляции и исполнения	eval, execfile, reload, __import__, compile
Функции ввода-вывода	input, raw_input, open
Функции для работы с атрибутами	getattr, setattr, delattr, hasattr
Функции-«украшатели» методов классов	staticmethod, classmethod, property
Прочие функции	buffer, slice

КЛАССЫ

Оператор class применяется для определения объектов новых типов и для объектно-ориентированного программирования.

В классе метод определяется с помощью оператора def. Первый параметр каждого метода всегда ссылается на сам объект.

В соответствии с общепринятым соглашением для этого параметра применяется имя self.

Все операции, затрагивающие атрибуты объекта, должны явно ссылаться на переменную self.

Методы с ведущими и конечными двойными подчеркиваниями являются специальными методами.

Метод (конструктор) __init__ применяется для инициализации объекта при его создании.

ИСКЛЮЧЕНИЯ

Если в программе возникает ошибка, то активизируется исключение и появляется сообщение об ошибке, как в следующем примере:

В сообщении об ошибке указан тип возникшей ошибки, а также место ее возникновения. Обычно ошибки приводят к аварийному завершению программы. Однако можно перехватывать и обрабатывать исключения с помощью операторов try и except.

При возникновении ошибки IOError подробные сведения о причинах ошибки помещаются в переменную e и управление передается коду в блоке except. При возникновении исключения какого-то другого вида оно передается во включающий блок кода (если он есть). Если ошибки не возникают, код в блоке except игнорируется. Для сообщения об исключении используется оператор raise.

ЧИСЛОВЫЕ И СТРОКОВЫЕ ФУНКЦИИ

Таблица 5

Функции	Описание
1	2
abs(x)	Модуль числа x
divmod(x, y)	Частное и остаток от деления

Окончание табл. 5

1	2
pow(x, y[,m])	Возведение x в степень y
round(n[, z])	Округление чисел до заданного знака после точки
ord(s)	Функция возвращает код заданного символа в строке
chr(n)	Возвращает строку с символом с заданным кодом
len(s)	Возвращает число элементов последовательности
oct(n), hex(n)	Возвращают строку с восьмеричным или шестнадцатеричным представлением целого числа n
cmp(x, y)	Сравнение двух значений. Результат: отрицательный, ноль или положительный, в зависимости от результата сравнения
Unicode(s[, encoding[, errors]])	Создает Unicode-объект, соответствующий строке s в заданной кодировке encoding

ФУНКЦИИ ОБРАБОТКИ ДАННЫХ

В программе Python предопределены функции генерации массива чисел. Пример с функциями range() и enumerate().

ФУНКЦИИ ОПРЕДЕЛЕНИЯ СВОЙСТВ

Функции определения свойств обеспечивают доступ к некоторым встроенным атрибутам объектов и другим свойствам.

ФУНКЦИИ ВВОДА-ВЫВОДА

Функции input() и raw_input() используются для ввода со стандартного ввода. Функция open() служит для открытия файла по имени для чтения, записи или изменения. Функция принимает три аргумента.

Первые два – имя файла (путь к файлу), режим открытия ("r" – чтение, "w" – запись, "a" – добавление или "w+", "a+", "r+" – изменение; также может прибавляться «t», что обозначает текстовый файл и имеет значение только на платформе Windows).

Третий аргумент указывает режим буферизации: 0 без буферизации; 1 построчная буферизация; больше 1 буфер указанного размера в байтах.

Выполнение работы:

1. Разработайте приложение, следуя указанным рекомендациям с широким применением элементов управления.

Практическая работа №3
Составление отчета по выполненной работе

(3 часа)

Цель работы: изучить требования и методику составления отчета, выполнить и представить отчет

Технические средства и программное обеспечение:

1. IBM-PC или совместимый компьютер;
2. Операционная система Microsoft Windows;
3. Пакет офисных программ Microsoft Office;
4. Microsoft Visual Studio Community.

Теоретические сведения:

Отчет по работе оформляется на бумаге стандартного формата (формат А4).

Тема отчета формулируется на основе темы выпускной квалификационной работы.

Коллективное составление и сдача отчетов не допускается.

Отчет по практическим работам должен быть выполнен в текстовом редакторе Microsoft Word 2010 или выше и содержать: титульный лист (рис.), название темы работы, цели работы, перечень технических и программных средств, необходимых для выполнения практической работы; краткое описание исследуемого вопроса; алгоритм программы; исходные данные варианта; распечатку полученных в ходе расчета значений; выводы, содержащие анализ проведенной работы.

В выводах дается краткое объяснение сущности полученных результатов. Выводы должны быть краткими и отвечать на вопросы, поставленные в практической работе.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИМИ

ФГБОУ ВО «Воронежский государственный
технический университет»

Факультет машиностроения и аэрокосмической техники

Кафедра автоматизированного оборудования
машиностроительного производства

ОТЧЕТ

По выполнению практических работ по дисциплине
«Автоматизированные методы проектирования
технологических процессов»
на тему
« »

Разработал студент(ка) гр. мМП-

«___» _____

Принял ст. преподаватель кафедры «Автоматизированного оборудования машиностроительного производства»

«___» _____ С. Л. Новокщенов

Воронеж 202_

Рис. 15. Пример оформления титульного листа отчета

Выполнение работы:

1. По указанию преподавателя оформить отчет по проделанным работам с соблюдением требований, установленных в теоретических сведениях практической работы № 3.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Автоматизация проектирования технологических процессов в машиностроении / под ред. Н. М. Капустина. — М.: Машиностроение, 1985.

2. Петухов, А. В. и др. Системы автоматизированного проектирования технологических процессов: учебное пособие / А. В. Петухов и др. – Беларусь, Гомел. гос.техн. ун-т им. П.О Сухого, 2011. – 144 с.

3. Петухов А. В., Мельников Д. В. Системы автоматизированного проектирования технологических процессов: лабораторный практикум. – Беларусь, г. Гомель, ГГТУ имени П.О. Сухого, 2010. – 190 с.

4. Леонтьев. Б. П. Методы и алгоритмы компьютерного инженерно-технологического проектирования химических производств: монография / Б. П. Леонтьев. — Тольятти: ТГУС, 2010. – 130 с.

5. Буйначев, С. К. и др. Основы программирования на языке Python : учебное пособие / С. К. Буйначев, Н. Ю. Боклаг. – Екатеринбург: изд-во Урал. ун-та, 2014. – 91, [1] с.

ОГЛАВЛЕНИЕ

АВТОМАТИЗИРОВАННЫЕ МЕТОДЫ ПРОЕКТИРОВАНИЯ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению практических работ
студентами направления подготовки
15.04.01 «Машиностроение»
(программа магистерской подготовки «Обеспечение качественно-точностных характеристик при изготовлении изделий в автоматизированном машиностроительном производстве»)
очной и заочной форм обучения

Составитель
Новокщенов Сергей Леонидович

В авторской редакции

Подписано к изданию 03.06.2022.
Уч.-изд. л. 2,0

ФГБОУ ВО «Воронежский государственный технический университет»
394006 Воронеж, 20-летия Октября, 84