УДК 621.91.01
DOI: 10.17277/vestnik.2022.04.pp.674-684

# РАЗРАБОТКА АЛГОРИТМА РАСЧЕТА ПРИПУСКА НА МЕХАНИЧЕСКУЮ ОБРАБОТКУ ДЕТАЛЕЙ ДЛЯ СРЕДЫ ПРОГРАММИРОВАНИЯ PYTHON

**Н. В. Бондаренко, М. В. Соколов**

*Кафедра «Компьютерно-интегрированные системы в машиностроении», msok68@mail.ru; ФГБОУ ВО «ТГТУ», Тамбов, Россия*

**Ключевые слова:** алгоритм; база данных; межоперационные переходы; припуск на механическую обработку деталей; программа.

**Аннотация:** Рассмотрена и решена задача расчета рационального припуска с использованием среды программирования Python. Разработаны алгоритм и программа для расчета рационального припуска и определения необходимого количества переходов для достижения заданных конструкторской документацией параметров шероховатости поверхности и точности размеров.

---

Одной из проблем современного машиностроения является определение припуска, который необходимо снять с заготовки, для того чтобы обеспечить необходимые геометрическую точность и чистоту поверхности детали. Формулы для расчета припуска представлены в работе [1].

Определение рациональной толщины припуска на обработку является важной технико-экономической задачей. Маленький припуск может не обеспечить заданных параметров шероховатости детали. Кроме того, на заготовке может сохраниться дефектный слой с предыдущей операции, что приведет к браку. Большие припуски могут привести к экономическим потерям, так как придется снимать много материала для достижения заданных параметров.

Для решения данной проблемы применяют два метода: аналитический и справочный. Первый позволяет точно рассчитать максимальный, минимальный и номинальный припуски на операцию, но при этом требует времени для выполнения расчетов и их проверки, особенно, когда деталь имеет сложную конфигурацию. Поэтому наибольшее распространение получил справочный метод, определяющий величину припуска по машиностроительным справочникам, которая составляется на основе данных, полученных на предприятиях. Данный метод более быстрый, но менее точный. При разработке технологического процесса инженеры-технологи чаще используют свои знания и личный опыт при назначении последовательности операций механической обработки.

Однако в современном машиностроении пользоваться справочниками не всегда удобно. Поэтому в качестве еще одного варианта для определения припуска могут применяться специальные программы, которые по исходным данным способны рассчитать припуск. Тем не менее программы имеют ряд недостатков.

Во-первых, подобные программы нуждаются в поддержке и обновлениях, так как исходные данные для расчетов программы получают не только от пользователя, но и от баз данных, встроенных в программу. Если базы данных не обновлять своевременно, то программа будет выдавать неточные результаты расчетов, которые не соответствуют реальным условиям механической обработки.

Во-вторых, недостаточная кросс-платформенность программы. Для того чтобы программа работала, ее необходимо установить на компьютер или смартфон со всеми необходимыми библиотеками или базами данных, но из-за большого их количества занимаемая физическая память программы оказывается огромной, что затрудняет ее установку на слабых устройствах. Эту проблему можно решить при помощи облачных вычислений, когда все базы данных хранятся на сервере, и там же происходят вычисления, при этом не следует устанавливать программу, а для ее активации достаточно будет зайти на сайт разработчика.

К преимуществам программы можно отнести ее высокую скорость работы, при рациональной реализации – кросс-платформенность, применение современных технологий для реализации, а, также перспективу ее развития, так как в дальнейшем к этой программе можно подключать дополнительные модули, расширяющие ее возможности.

Вопросы обеспечения качества изделий на стадии технологической подготовки производства, рассмотренные при проектировании системы поддержки принятия решений выбора режимных и конструктивных параметров, изложены в работах [2 – 4].

Для определения припуска инженер должен знать, какая заготовка будет использоваться при изготовлении детали. Как правило, материал для заготовки подбирает конструктор, используя справочную литературу – сортаменты, ГОСТы, ОСТы и др.

В качестве первого этапа определения припуска выступает определение необходимого количества операций. Для этого определяют коэффициент уточнения $\varepsilon_i$

$$\varepsilon_i = T_{i-1}/T_i\,, \tag{1}$$

где $T_{i-1}$, $T_i$ – допуски на размер, полученные соответственно на предыдущей и выполняемой технологических операциях.

Так как для получения детали из заготовки, заготовка должна пройти через несколько операций, определяют общий коэффициент уточнения $\varepsilon_o$

$$\varepsilon_o = \prod_{i=1}^{i=k} i, \tag{2}$$

где $k$ – количество операций.

Далее определяют требуемое уточнение, которое необходимо обеспечить при обработке:

$$\varepsilon_{TO} = T_з/T_д, \tag{3}$$

где $T_з$, $T_д$ – допуски на размеры соответственно заготовки и детали.

Сравнивают $\varepsilon_o$ и $\varepsilon_{TO}$. Если количество операций подобрано правильно, то

$$\varepsilon_o \le \varepsilon_{TO}. \tag{4}$$

Выполнение неравенства (4) гарантирует достижение требуемой точности (3). Затем переходят к определению припуска.

Расчеты ведут по трем формулам – для определения минимального, номинального и максимального припусков (формулы могут различаться, в зависимости от операции и типа обрабатываемой поверхности):

$$Z_{i\min} = Rz_{(i-1)} + h_{i-1} + \Delta_{i-1} + \varepsilon_{i-1}; \tag{5}$$

$$Z_i = Z_{i\min} + eiD_{i-1} + esD_i; \tag{6}$$

$$Z_{i\max} = Z_{i\min} + ITD_{i-1} + ITD_i, \tag{7}$$

где $Rz$ – высота неровностей поверхности; $h$ – глубина дефектного слоя; Δ – суммарные отклонения расположения поверхностей; ε – погрешность установки заготовки; $eiD_{i-1}$, $esD_i$ – предельные отклонения размеров соответственно на предшествующем и выполняемом переходах; $ITD_{i-1}$, $ITD_i$ – допуски на получаемый размер соответственно на предшествующем и выполняемом переходах.

Параметры $Rz$ и $h$ можно определить как по справочной литературе, так и практическими методами, например, измерив шероховатость поверхности детали на профилометре или сравнивая с образцами шероховатости. Коэффициент Δ чаще всего выбирают с использованием справочной литературы, а коэффициент ε можно измерить непосредственно на заготовке, установленной в приспособление на станке.

Когда припуск на механическую обработку рассчитан, определяют межпереходные размеры. Формула для определения межоперационных размеров при обработке поверхностей вращения имеет следующий вид:

$$D_{i-1} = D_i - Z_i, \tag{8}$$

где $D_i$, $D_{i-1}$ – номинальные размеры соответственно на $i$-м и предшествующем переходах; $Z_i$ – номинальный припуск на $i$-м переходе.

Определив межпереходные размеры толщину припуска на обработку, можно, для наглядности, изобразить их расположение на эскизе (рис. 1).

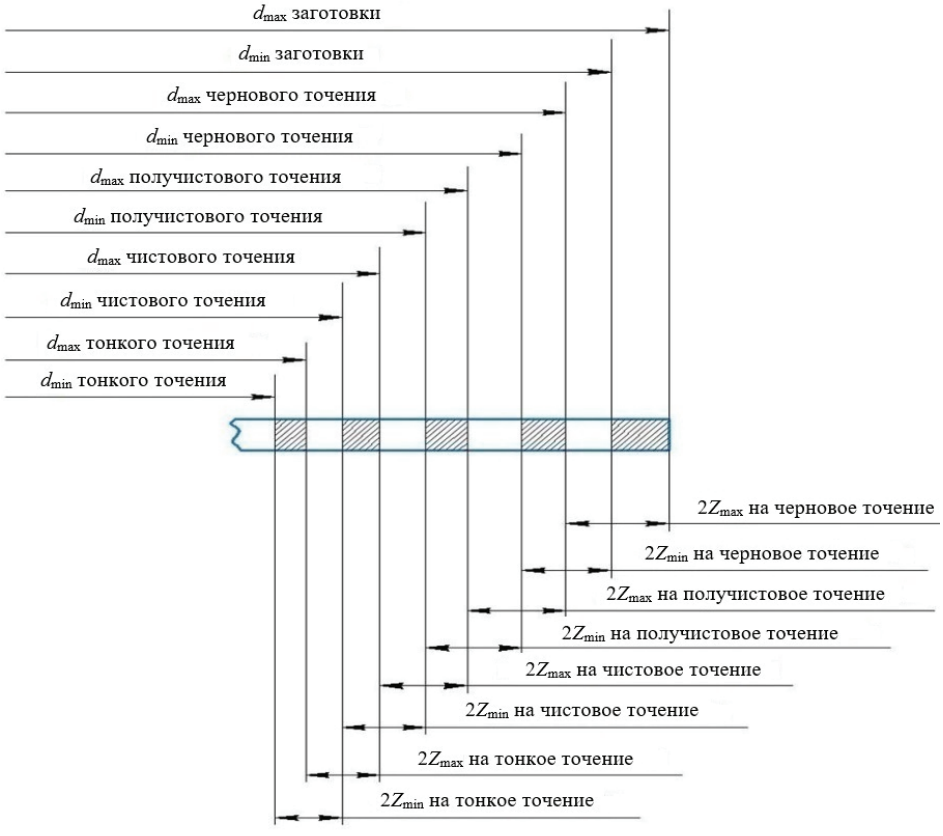


**Рис. 1. Расположение припусков и межоперационных размеров для токарной обработки внешней цилиндрической поверхности**

За последние двадцать лет отрасль «Машиностроение» сделала большой скачок в развитии. Чтобы поддержать эффективность и конкурентоспособность, предприятия машиностроения постоянно внедряют инновации и цифровые технологии. Однако, несмотря на то, что все больше и больше появляется станков с числовым программным управлением (**ЧПУ**) с относительно простым пользовательским интерфейсом, которые позволяют более экономично использовать рабочее пространство и проще настраиваются на обработку деталей, множество инженерных задач все еще остается нерешенным.

Так как основными задачами машиностроения являются непрерывное повышение качества машин и оборудования и совершенствование роста производительности труда на предприятиях, то автоматизация инженерных расчетов позволит повысить производительность труда за счет того, что инженер сможет быстрее разработать технологический процесс и выполнить остальные необходимые действия.

В России большой популярностью пользуются системы автоматизированного проектирования компании «Аскон», в частности «Компас-3D» и «Вертикаль». Система «Компас-3D» помогает создавать чертежи и модели деталей, а «Вертикаль» – техпроцессы и заполнять маршрутные, операционные и эскизные карты. Если первая достаточно проста в освоении, то вторую придется изучать. Система «Вертикаль» работает с библиотеками или базами данных, которые можно создавать самостоятельно. Однако «Компас-3D» и «Вертикаль» не могут рассчитывать припуски и режимы резания для операций, а значит, разработка программы, способной выполнять такие расчеты, актуальна.

Разработка любой программы происходит в среде программирования. Существует множество различных языков программирования, некоторые из них больше подходят для создания сайтов (PHP), другие – универсальные, но сложные в усвоении (C++, C#) – они подходят для написания сложных программ, которые могут работать с трехмерными объектами. Но для создания программ для инженерных расчетов использование вышеперечисленных языков не обязательно.

Среди программистов принято считать Python низкоуровневым языком программирования, подходящим новичкам для написания простых программ. Однако функционал этого языка гораздо шире. Он действительно простой – освоить Python можно за две-три недели, при том, если пользователь совершит ошибку, среда программирования сразу укажет место и причину ошибки, что позволяет сразу же ее исправить. Кроме того, для Python существует множество библиотек, которые являются бесплатными (как и он сам), что позволяет проводить сложные вычисления, работать с 3D-моделями и базами данных.

Кроме того, Python позволяет работать с нейронными сетями. Нейронная сеть (нейросеть) – математическая модель (а также ее программное или аппаратное воплощение), построенная по принципу организации и функционирования биологических нейронных сетей – сетей нервных клеток живого организма [5]. В машиностроении нейросети пока не применяют, однако, скорее всего, дальнейшее развитие нейросетей приведет к тому, что они станут применяться повсеместно [4].

Приведем примеры наиболее известных и популярных библиотек:

– SciPy – библиотека, используемая учеными, аналитиками и инженерами, занимающимися научными и техническими вычислениями. Она содержит модули для решения задач оптимизации, линейной алгебры, интерполяции специальных функций, а также обработки сигналов и изображений [6];

– Matplotlib – библиотека Python для построения 2D-графиков, которая позволяет получать изображения в интерактивных кроссплатформенных средах. С ее помощью создают графики, гистограммы, спектры мощности, диаграммы ошибок и диаграммы разброса [6];

– OpenPyXL – библиотека, позволяющая Python считывать, изменять и использовать в расчетах данные из программы MSExcel, на основе которой можно создавать простые базы данных в виде таблиц;

– OpenCV – библиотека, рассчитанная на работу с «машинным зрением», ее часто используют при обучении нейросетей работе с изображениями;

– Tensorflow и Keras – мощные библиотеки AI (ИИ) для написания нейросетей. Используя их, можно разработать абсолютно любую нейросеть – от простейшего распознавания рукописных цифр до управления автомобилем в режиме реального времени [7];

– NumPy – один из самых основных пакетов в Python – универсальный пакет для обработки массивов. Он представляет высокопроизводительные объекты многомерных массивов и инструменты для работы с ними [7].

Кроме того, на Python уже существуют программы, позволяющие решать разные инженерные задачи, например FreeCAD.

FreeCAD – бесплатная программа для параметрического трехмерного компьютерного проектирования с поддержкой метода конечных элементов. Она предназначена для машиностроения, но расширяется до более широкого круга применений, включая архитектуру или электротехнику. Python используется в качестве языка сценариев внутри FreeCAD. Пользователи могут самостоятельно расширять с его помощью функции приложения [6].

Несмотря на все преимущества Python, у него есть и недостатки: для работы со сложными библиотеками, к которым относятся Tensorflow и Keras, необходимо четкое понимание их работы, чтобы избежать ошибок при написании кода программы, но в российском сегменте Интернета очень мало справочной информации о работе библиотек, и существуют лишь ограниченные примеры их использования, что не позволяет самостоятельно разобраться во всех нюансах их работы. В то же время, некоторые библиотеки могут конфликтовать друг с другом из-за несовместимости версий, что ведет к сбоям во время работы программы. Однако для начинающих программистов Python является наиболее простым и удобным языком программирования, поэтому для разработки программы для расчета припуска под механическую обработку будем использовать именно его.

В первую очередь необходимо установить своеобразные «границы», в рамках которых разрабатывается программа. Так, например, нет необходимости разрабатывать интерфейс программы – это займет долгое время, а кроме того, потребует переработать код программы с учетом расположения виджетов – кнопок, окон, ползунков и прочих элементов. Первостепенной задачей является написание рабочего кода, который будет выполнять расчеты и выводить результат, а разработать интерфейс можно будет потом. В ходе практических исследований разработан прототип программы, который в дальнейшем будет обрастать новыми возможностями.

Так как программа работает с табличными значениями, необходимо их перенести в отдельный электронный формат. Обычно в таких случаях используют систему управления базами данных (**СУБД**), например – Microsoft Access, Database.NET, MySQLWorkbench и др. [8]. Но в данном случае нет необходимости их использовать, так как можно пойти по более легкому пути и создать файл формата .xlxs, то есть таблицу Microsoft Excel. У Python есть отдельная библиотека для работы с такими файлами, которая будет использоваться. Это позволит упростить задачу и сэкономить время при разработке программы, так как Microsoft Excel, в отличии от СУБД, не так сложен в изучении.

На данном этапе необходимо определить параметры $Rz$, $h$ и $\Delta$, указанные в таблицах 2.5 – 2.13 [1]. Для использования их в Python, создадим такие же таблицы в Excel с данными о прокате (табл. 1) и погрешности $\Delta_к$ (табл. 2).

Таблица 1

**Качество поверхности сортового проката**

| Диаметр проката, мм | Точность проката, мкм | | | | | |
|---|---|---|---|---|---|---|
| | Высокая | | Повышенная | | Обычная | |
| | *Rz* | *h* | *Rz* | *h* | *Rz* | *h* |
| До 30 | 63 | 50 | 80 | 100 | 125 | 150 |
| Свыше: | | | | | | |
| 30 до 80 | 100 | 75 | 125 | 150 | 160 | 250 |
| 80 до 180 | 125 | 100 | 160 | 200 | 200 | 300 |
| 180 до 250 | 200 | 200 | 250 | 300 | 320 | 400 |

Таблица 2

**Кривизна профиля сортового проката**

| Характеристика проката, мм | Длина проката, мкм на 1 мм | | | | |
|---|---|---|---|---|---|
| | До 120 | 120…180 | 180…315 | 315…400 | 400…500 |
| Без правки при точности проката: | | | | | |
| обычной | 0,5 | 1,0 | 1,5 | 2,0 | 2,5 |
| повышенной | 0,2 | 0,4 | 0,6 | 0,8 | 1,0 |
| высокой | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 |

Чтобы функционал программы не был ограничен одним лишь прокатом, необходимо создать больше таблиц с информацией о разных заготовках.

Параметр Δ, согласно [1], определяется по разным формулам, в зависимости от способа закрепления заготовки. Большая часть формул включает в себя два параметра: $\Delta_к$ и $\Delta_{см}$ – кривизну заготовки и отклонение от соосности соответственно. Сами формулы будут прописаны в коде программы отдельно, а параметры $\Delta_к$ и $\Delta_{см}$ будем использовать как переменные, значения которым зададим из таблицы.

Далее библиотеку OpenPyXl подключаем к программе в 3-й строке (рис. 2).

Затем импортируем из нее функцию load_workbook, позволяющую работать с конкретным файлом, который подключаем в 6-й строке, а для подтверждения подключения прописываем строку 7: print (wb.sheetnames) – эта команда выведет номер активный страницы на экран. Восьмая строка делает данную страницу активной, что позволит вносить и извлекать данные из ранее созданной таблицы.

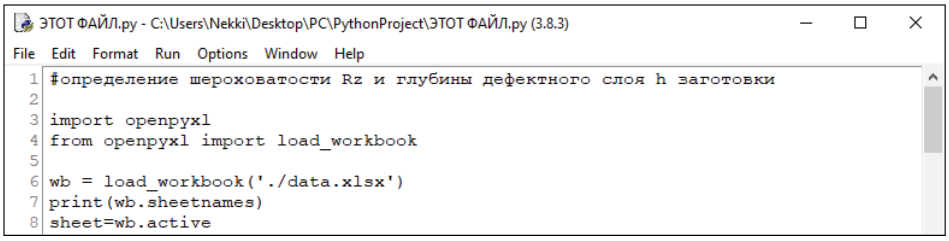


**Рис. 2. Подключение библиотеки OpenPyXl**

После запуска программы происходит проверка подключения (рис. 3). Программа вывела номер страницы, что означает следующее: прописанный выше код был правильным. Затем начинается выполнение самого кода, но прежде чем приступить к расчетам, необходимо добавить цикл, который не будет выключать программу после окончания расчетов, чтобы выполнять разные расчеты без ее перезапуска (рис. 4, *а*).

В строке 10 создается переменная index типа integer, хранящая в себе целочисленное значение, в данном случае она равна нолю, а в строке 11 – сам цикл while, который дословно переводится как «пока». Это означает, что пока index = 0, программа будет работать. Когда же пользователь захочет прекратить работать с программой и закрыть ее, ему будет предложен выбор (рис. 4, *б*). Если пользователь выберет ответ «Да», то программа продолжит работать, если «Нет» – программа закроется. В дальнейшем будет реализована еще одна функция, которая позволит пользователю сохранить расчеты в отдельный файл при закрытии программы.

Далее реализуем основной код программы. Сначала узнаем у пользователя, для какой операции проводятся расчеты, и в зависимости от ответа будем использовать соответствующие формулы (рис. 5). В строке 13 создаем переменную operaciya типа string, которая может хранить в себе слово или фразу (см. рис. 5, *а*). Для того чтобы присвоить данной переменной значение, используется команда input, позволяющая пользователю напечатать нужное значение в терминале. Но в данном случае пользователю нет необходимости писать слово – варианты операций уже предложены на экране и все они имеют свой номер, достаточно лишь указать его. Примечательно, что в данном случае использовать переменную типа integer (int) нельзя, так как команда input не работает с этим типом переменной.

Затем создаем цикл if-elif-else, который работает по следующему принципу: команда if получает заданное условие, а затем выполняет действие, соответствующее этому условию (см. рис. 5, *б*). В данном случае после if стоит условие: переменная operaciya должна хранить значение 1. Если это условие выполняется, то расчет происходит по формуле, написанной в 587-й строке, после чего выводится

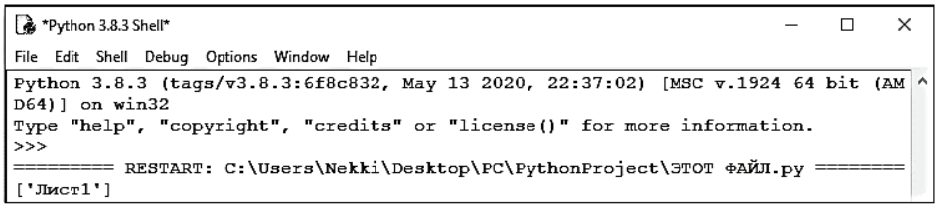


**Рис. 3. Проверка подключения библиотеки OpenPyXl**

```
10 index=int=0
11 while index==0:
```

*а*)

```
651 pr=str(input("Продолжить?(Да/Нет)"))
652 if pr=="Да":
653     index=0
654 else:
655     index=1
```

*б*)

**Рис. 4. Начало (а) и конец (б) цикла программы**

```
13    operaciya = str(input("Укажите тип операции:\n"
14                          "[1]Токарная (только внешняя или только внутренняя обработка)\n"
15                          "[2]Токарная (одновременная обработка внутренней и внешней поверхностей)\n"
16                          "[3]Фрезерная\n"))
```

*а*)

```
585     if operaciya=="1":
586         Zmin = float
587         Zmin = 2*(Rz+h+Delta+ε)
588         print("2Zmin = ",Zmin,"мкм.")
589     elif operaciya=="2":
590         Zmin = float
591         Zmin = 2*(Rz+h)+2*(((Delta**2)+(ε**2))**0,5)*(Rz+h+Delta+ε)
592         print(Zmin)
593     elif operaciya=="3":
594         Zmin = float
595         Zmin = Rz+h+Delta+ε
596         print(Zmin)
597     else:
598         print("Данные отсутствуют!")
```

*б*)

**Рис. 5. Определение типа операции (*а*) и припуска для конкретно указанной операции (*б*)**

результат. Если переменная operaciya хранит значение 2, то программа пропускает строки 586, 587, 588 и начинает выполнять строки 590, 591, 592 и так далее, до тех пор, пока цикл не будет закрыт командой else (в данном случае эта команда просто прекратит выполнение всей программы).

Затем, когда пользователь выбрал операцию, он должен указать заготовку и некоторые ее характеристики – точность для проката, массу и способ изготовления для штамповки, диаметр, массу и способ изготовления для поковки. Данную информацию можно получить либо с основной надписи чертежа, либо у завода-изготовителя заготовки, либо просто замерив необходимые параметры у заготовки (последнее необходимо, если заготовкой является литая деталь). После указания необходимых параметров будут созданы две переменные типа float, которые содержат в себе дробные значения: шероховатость и глубина дефектного слоя материала – *Rz* и *h*. Таким образом, определены два параметра для расчета припуска.

На рисунке 6 в качестве примера приведен отрывок кода для подбора шероховатости и глубины дефектного слоя для заготовки из проката высокой, повышенной и обычной точности. Вне зависимости от выбранной точности программа попросит пользователя указать диаметр проката, от которого уже будет зависеть значение, присваиваемое *Rz* и *h*. При выполнении кода программа выведет присвоенные значения (рис. 7).

Как можно видеть, если пользователь укажет, что заготовка изготовлена из проката высокой точности диаметром 85 мм, то шероховатость поверхности заготовки составит 125 мкм, а глубина дефектного слоя – 100 мкм. Для того чтобы убедиться в правильности выбранных параметров и работоспособности программы, сравним полученные значения с табличными [1, табл. 2.5], которые ранее перенесены в Microsoft Excel (см. табл. 1).

По результатам сравнительного анализа расчетных и табличных значений для диаметра проката 80…180 мм при высокой точности шероховатость составит 125 мкм, а глубина дефектного слоя – 100 мкм, что соответствует данным, полученным в ходе выполнения программы. Таким образом, удалось реализовать часть кода программы, который определяет *Rz* и *h* для одного типа заготовок.

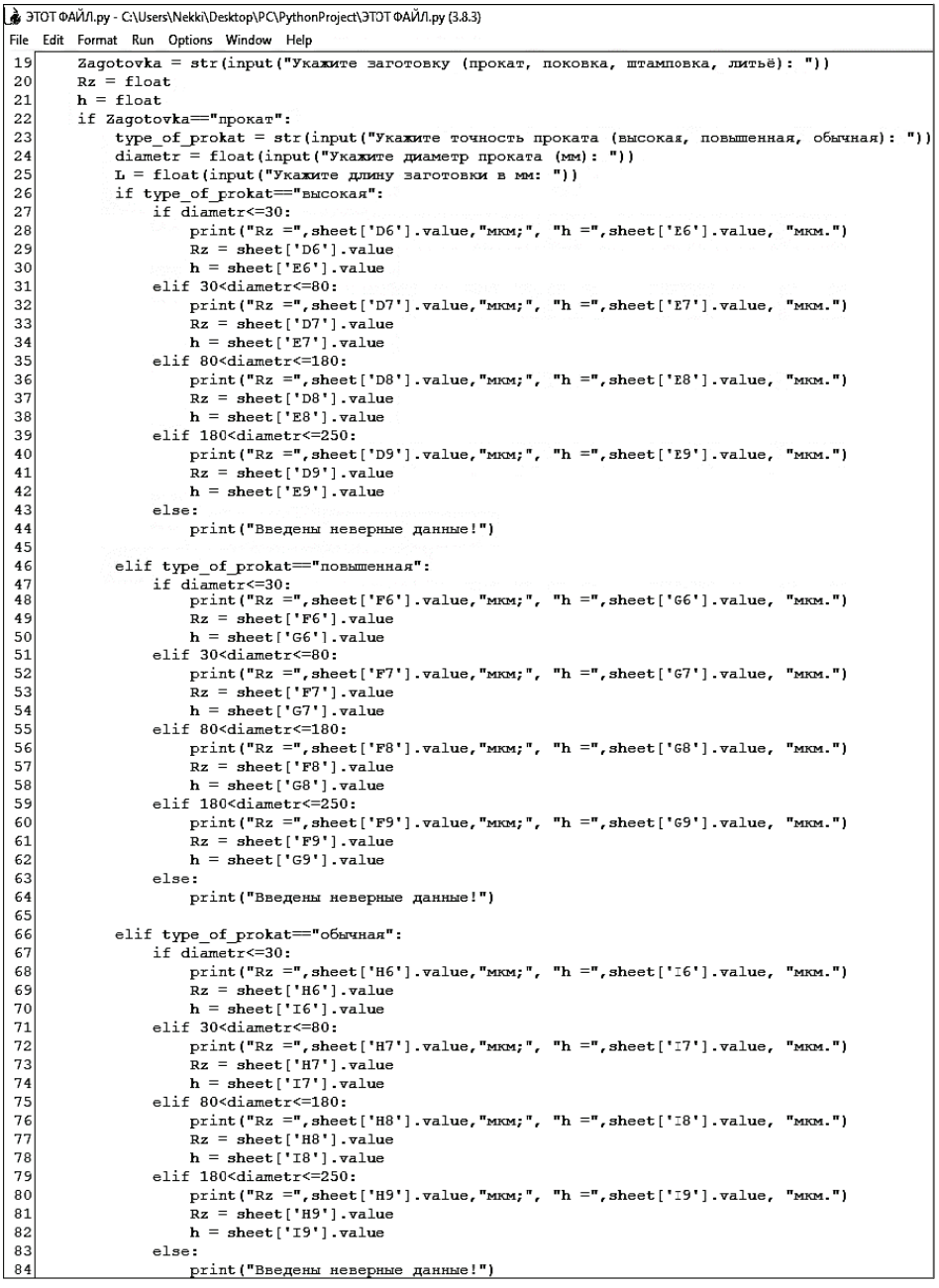


**Рис. 6. Определение шероховатости и глубины дефектного слоя заготовки**

```
Укажите заготовку (прокат, поковка, штамповка, литьё): прокат
Укажите точность проката (высокая, повышенная, обычная): высокая
Укажите диаметр проката (мм): 85
Укажите длину заготовки в мм: 100
Rz = 125 мкм; h = 100 мкм.
```

**Рис. 7. Определение шероховатости и глубины дефектного слоя заготовки в зависимости от введенных параметров**

*Список литературы*

1. Расчет припусков и межпереходных размеров в машиностроении / Я. М. Радкевич, В. А. Тимирязев, А. Г. Схиртладзе, М. С. Островский ; под ред. В. А. Тимирязева. – Изд. 2-е, стер. – М. : Высш. школа, 2007. – 272 с.
2. Концепция создания системы автоматизированного проектирования процессов резания в технологии машиностроения / С. И. Пестрецов, К. А. Алтунин, М. В. Соколов, В. Г. Однолько. – М. : Спектр, 2012. – 212 с.
3. Altunin, K. A. Development of Information Support for Intelligent Cad of Cutting Processes / K. A. Altunin, M. V. Sokolov // Advanced Materials and Technologies. – 2017. – No. 2. – P. 67 – 77. doi: 10.17277/amt.2017.02.pp.067-077
4. Алтунин, К. А. Применение нейронных сетей для моделирования процесса токарной обработки / К. А. Алтунин, М. В. Соколов // Вестн. Тамб. гос. техн. ун-та. – 2016. – Т. 22, № 1. – С. 122 – 133. doi: 10.17277/vestnik.2016.01.pp.122-133
5. Кравченко, С. Примеры использования Python, вдохновляющие на его изучение. – Текст : электронный / С. Кравченко // Proglib. – URL : https://proglib.io/p/primery-ispolzovaniya-python-vdohnovlyayushchie-na-ego-izuchenie-2021-02-21 (дата обращения: 10.09.2022).
6. Сайт FreeCAD. – URL : https://www.freecadweb.org/index.php?lang=ru (дата обращения: 10.09.2022).
7. Топ-10 библиотек Python для Data Science. – Текст : электронный // DataStart. – URL : https://datastart.ru/blog/read/top-10-bibliotek-python-dlya-data-science (дата обращения: 10.09.2022).
8. Программы для работы с базами данных. – Текст : электронный // Lumpics.ru. – URL : https://lumpics.ru/database-software (дата обращения: 10.09.2022).

---

## Development of an Algorithm for Calculation of the Allowance for Mechanical Processing of Parts for the Python Programming Environment

**N. V. Bondarenko, M. V. Sokolov**

*Department of Computer-Integrated Systems in Mechanical Engineering, msok68@mail.ru; TSTU, Tambov, Russia*

**Keywords:** algorithm; database; interoperational transitions; allowance for machining parts; program.

**Abstract:** The problem of calculating a rational allowance using the Python programming environment is considered and solved. An algorithm and a program have been developed for calculating the rational allowance and determining the required number of transitions to achieve the parameters of surface roughness and dimensional accuracy specified by the design documentation.

*References*

1. Radkevich Ya.M., Timiryazev V.A. [Ed.], Skhirtladze A.G., Ostrovskiy M.S. *Raschet pripuskov i mezhperekhodnykh razmerov v mashinostroyenii* [Calculation of allowances and intertransitional dimensions in mechanical engineering], Moscow: Vysshaya shkola, 2007, 272 p. (In Russ.)

2. Pestretsov S.I., Altunin K.A., Sokolov M.V., Odnol'ko V.G. *Kontseptsiya sozdaniya sistemy avtomatizirovannogo proyektirovaniya protsessov rezaniya v tekhnologii mashinostroyeniya* [The concept of creating a system for automated design of cutting processes in engineering technology], Moscow: Spektr, 2012, 212 p. (In Russ.)

3. Altunin K.A., Sokolov M.V. Development of Information Support for Intelligent Cad of Cutting Processes, *Advanced Materials and Technologies*, 2017, no. 2, pp. 67-77, doi: 10.17277/amt.2017.02.pp.067-077 (In Eng., abstract in Russ.)

4. Altunin K.A., Sokolov M.V. [Application of neural networks for modeling the process of turning], *Transactions of the Tambov State Technical University*, 2016, vol. 22, no. 1, pp. 122-133, doi: 10.17277/vestnik.2016.01.pp.122-133(In Russ., abstract in Eng.)

5. https://proglib.io/p/primery-ispolzovaniya-python-vdohnovlyayushchie-na-ego-izuchenie-2021-02-21 (accessed 10 September 2022).

6. https://www.freecadweb.org/index.php?lang=ru (accessed 10 September 2022).

7. https://datastart.ru/blog/read/top-10-bibliotek-python-dlya-data-science (accessed 10 September 2022).

8. https://lumpics.ru/database-software (accessed 10 September 2022).

---

## Entwicklung eines Algorithmus für Berechnung der Zulage zur mechanischen Bearbeitung von Teilen für die Python-Programmierumgebung

**Zusammenfassung:** Es ist das Problem der Berechnung eines rationalen Aufmaßes mit der Programmierumgebung Python betrachtet und gelöst. Es sind ein Algorithmus und ein Programm zur Berechnung des rationalen Aufmaßes und zur Bestimmung der erforderlichen Anzahl von Übergängen entwickelt, um die in der Konstruktionsdokumentation angegebenen Parameter der Oberflächenrauheit und Maßhaltigkeit zu erreichen.

---

## Élaboration d'un algorithme de calcul de l'allocation sur l'usinage de pièces pour l'environnement de programmation Python

**Résumé:** Est examiné et résolu le problème du calcul de l'allocation rationnelle en utilisant l'environnement de programmation Python. Sont élaborés un algorithme et un programme d'allocation rationnelle pour calculer et déterminer le nombre nécessaire de transitions pour atteindre les paramètres de rugosité de surface et de précision dimensionnelle spécifiés par la documentation de conception.

---

**Авторы:** ***Бондаренко Никита Владимирович*** **–** магистрант; ***Соколов Михаил Владимирович*** – доктор технических наук, профессор кафедры «Компьютерно-интегрированные системы в машиностроении», ФГБОУ ВО «ТГТУ», Тамбов, Россия.