

**МОДЕРНИЗАЦИЯ АЛГОРИТМА РАСЧЕТА ПРИПУСКА
НА МЕХАНИЧЕСКУЮ ОБРАБОТКУ ДЕТАЛЕЙ
ДЛЯ СРЕДЫ ПРОГРАММИРОВАНИЯ PYTHON**

Н. В. Бондаренко, М. В. Соколов

*Кафедра «Компьютерно-интегрированные системы в машиностроении»,
nikbon.2015@yandex.ru, ФГБОУ ВО «ТГТУ», Тамбов, Россия*

Ключевые слова: алгоритм; допуск; межоперационные размеры; припуск на механическую обработку деталей.

Аннотация: Решена уточненная задача расчета рационального припуска с использованием среды программирования Python, модернизированы алгоритм и программа для расчета рационального припуска и определения необходимого количества переходов для достижения заданных конструкторской документацией параметров шероховатости поверхности и точности размеров. Дано описание разработанного алгоритма автоматического определения операций и назначения допуска, реализация которого в среде программирования Python позволяет выполнять расчеты для всех операций токарной обработки и подбирать их количество, что существенно упрощает работу технологом на стадии технологической подготовки машиностроительного производства.

Введение

В статье [1] рассмотрена проблема определения припуска, который необходимо снять с заготовки для того, чтобы обеспечить необходимую геометрическую точность детали, а также необходимую чистоту поверхности. Предложен способ ее решения с помощью программы с использованием языка программирования Python, которая позволила бы пользователю выполнить расчет припуска в автоматизированном режиме.

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].

В данной статье представлен существенно модернизированный алгоритм и программа для ЭВМ на языке Python.

Алгоритм автоматического определения операций на механическую обработку и назначения допусков на межоперационные размеры

На первом этапе модернизации алгоритма решена задача автоматического определения допусков на межоперационные размеры. Нередко для достижения заданной точности поверхности необходимо выполнить несколько операций: черновую, полустовую, чистовую и иногда финишную. При выполнении данных

операций получают межоперационные размеры (за исключением последней операции, на которой выполняется конечный размер детали). Технолог при разработке технологического процесса изготовления детали должен учитывать не только эти размеры, но и отклонения, которые могут возникнуть при их выполнении. Поэтому для этих размеров также необходимо назначать допуски.

Так как для расчета припуска используются параметры уточнения – общее и требуемое, которые напрямую зависят от допусков межоперационных размеров, то правильность подбора операций и их количества напрямую влияет на расчет припуска. Это видно в следующих формулах:

$$\varepsilon_i = T_{i-1}/T_i, \quad (1)$$

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

$$\varepsilon_0 = \prod_{i=1}^{i=k} \varepsilon_i, \quad (2)$$

где k – число операций;

$$\varepsilon_{T.O} = T_3/T_D, \quad (3)$$

где T_3 , T_D – допуски на размеры соответственно заготовки и детали;

$$\varepsilon_0 \geq \varepsilon_{T.O}. \quad (4)$$

Выполнение (4) подтверждает, что выбранных операций и их количества достаточно для достижения требуемой точности.

При разработке технологического процесса технолог может назначать допуски на межоперационные размеры самостоятельно с учетом своего опыта. Но компьютер сделать этого не может, а без этой функции программа может выполнить расчет припуска лишь для одной конкретной операции. Алгоритм назначения допуска позволил бы выполнять расчеты для всех операций и подбирать их количество, что существенно упростило задачу технологам.

На рисунке 1 программа выполняет расчет требуемого уточнения $\varepsilon_{T.O}$, используя данные, введенные пользователем. При этом программа определяет поля допусков размеров детали и заготовки в строках 19 и 26.

Затем выполняется определение качества для указанного диаметра детали (рис. 2).

Числовые значения, с которыми сравнивается параметр T_d , это поля допусков соответствующих качеств, согласно Единой системе допусков и посадок. Таким образом, происходит определение качества для любого заданного размера с любыми отклонениями, указанными пользователем. Когда он определен, можно

```

13 #получаем данные о диаметре, линейном размере, верхнем и нижнем отклонениях диаметра от пользователя:
14 D = float(input("Укажите диаметр детали в мм: "))
15 Vd = float(input("Укажите верхнее отклонение диаметра в мм: "))
16 Nd = float(input("Укажите нижнее отклонение диаметра в мм: "))
17 L = float(input("Укажите длину обрабатываемой поверхности в мм: "))
18 Td = float
19 Td = Vd-Nd
20 print("Допуск на диаметр детали: ", Td, "мм", "\n")
21
22 Z = float(input("Укажите размер заготовки в мм: "))
23 Vz = float(input("Укажите верхнее отклонение размера в мм: "))
24 Nz = float(input("Укажите нижнее отклонение размера в мм: "))
25 Tz = float
26 Tz = Vz-Nz
27 print("Допуск на размер заготовки: ", Tz, "мм", "\n")
28
29 eto = float
30 eto = Tz/Td
31 print("Требуемое уточнение: ", eto, "\n")

```

Рис. 1. Расчет требуемого уточнения

```

37 Kvalitet = int
38
39 if D<1:
40     print("Этот размер невозможно обработать на токарном станке!")
41 else:
42     #определяем качество
43     if 1<D<=3:
44         if Td<0.006:
45             Kvalitet = 5
46             print("Квалитет: ", Kvalitet)
47         elif 0.006<=Td<0.010:
48             Kvalitet = 6
49             print("Квалитет: ", Kvalitet)
50         elif 0.010<=Td<0.014:
51             Kvalitet = 7
52             print("Квалитет: ", Kvalitet)
53         elif 0.014<=Td<0.025:
54             Kvalitet = 8
55             print("Квалитет: ", Kvalitet)
56         elif 0.025<=Td<0.060:
57             Kvalitet = 9
58             print("Квалитет: ", Kvalitet)
59         elif 0.060<=Td<0.100:
60             Kvalitet = 11
61             print("Квалитет: ", Kvalitet)
62         elif 0.100<=Td<0.250:
63             Kvalitet = 12

```

Рис. 2. Определение качества

переходить к определению операций, но прежде необходимо создать переменные, которые будут хранить в себе параметры допусков.

На рисунке 3 показано создание переменных: $V_{\text{ОТКЛ}}$ – переменная, содержащее верхнее отклонение для соответствующей операции; $N_{\text{ОТКЛ}}$ – нижнее отклонение; T_{ch} , $T_{\text{p_ch}}$, T_{chist} – поля допуска для черновой, получистовой и чистовой операций соответственно; ε_{ch} , $\varepsilon_{\text{poluchern}}$, $\varepsilon_{\text{chist}}$, $\varepsilon_{\text{finish}}$ – уточнения для черновой, получистовой, чистовой и финишной операций соответственно. При этом параметр T_{finish} – переменная, которая хранила бы в себе поле допуска финишной операции – создавать нет смысла, так как на финишной операции обеспечивается размер детали в пределах его поля допуска, то есть T_d уже создан в 18 строке (см. рис. 1). Когда программа создала необходимые переменные, она приступает к выполнению алгоритма определения операций и назначения межоперационных допусков.

Рисунок 4 показывает, что созданные ранее переменные приобретают определенные значения с учетом качества размера детали и самого размера. Принцип назначения допусков заключается в следующем правиле: чем точнее конечный размер, тем больше требуется операций и тем точнее допуски на межоперационные размеры. По рассматриваемому алгоритму программа может назначить от

```

450 V_OTKL_chern_oper = float
451 N_OTKL_chern_oper = float
452 V_OTKL_poluchern_oper = float
453 N_OTKL_poluchern_oper = float
454 V_OTKL_chist_oper = float
455 N_OTKL_chist_oper = float
456 V_OTKL_finish_oper = float
457 N_OTKL_finish_oper = float
458
459 T_ch = float
460 T_p_ch = float
461 T_chist = float
462
463 ε_ch = float
464 ε_poluchern = float
465 ε_chist = float
466 ε_finish = float

```

Рис. 3. Создание переменных

одной до четырех операций, а также допуск для каждой операции и сразу выполнить проверку. Результат выполнения алгоритма представлен на рис. 5.

Программа определила качество и выполнила подбор операций, строка с цифрами «20.0 20.0» показывает значения требуемого и общего уточнений, из чего можно сделать вывод об их равенстве. О правильности выбора операций также говорит следующая строка «Операции определены верно». В случае, если бы операции были подобраны с ошибками, программа выведет сообщение: «Неизвестная ошибка!».

```

472 if Kvalitet <= 7:
473     if 1<D<=6:
474         V_OTKL_chern_oper = 0
475         N_OTKL_chern_oper = -0.2
476         T_ch = V_OTKL_chern_oper - N_OTKL_chern_oper
477         V_OTKL_poluchern_oper = 0
478         N_OTKL_poluchern_oper = -0.1
479         T_p_ch = V_OTKL_poluchern_oper - N_OTKL_poluchern_oper
480         V_OTKL_chist_oper = 0
481         N_OTKL_chist_oper = -0.02
482         T_chist = V_OTKL_chist_oper - N_OTKL_chist_oper
483         V_OTKL_finish_oper = Vd
484         N_OTKL_finish_oper = Nd
485         ε_ch = Tz/T_ch
486         ε_poluchern = T_ch/T_p_ch
487         ε_chist = T_p_ch/T_chist
488         ε_finish = T_chist/Td
489         εo = ε_ch*ε_poluchern*ε_chist*ε_finish
490         print(εo, εto)
491         if εo>=εto:
492             this_type_obrabotki = True
493             print("Операции определены верно.")
494             print("Необходимое количество операций: 4.")
495             print("Черновая, получистовая, чистовая, финишная.")
496         else:
497             this_type_obrabotki = False
498             print("Неизвестная ошибка!")

```

Рис. 4. Назначение допусков на межоперационные размеры

```

Укажите диаметр детали в мм: 50
Укажите верхнее отклонение диаметра в мм: 0
Укажите нижнее отклонение диаметра в мм: -0.01
Укажите длину обрабатываемой поверхности в мм: 35
Допуск на диаметр детали: 0.01 мм

Укажите размер заготовки в мм: 75
Укажите верхнее отклонение размера в мм: 0
Укажите нижнее отклонение размера в мм: -0.2
Допуск на размер заготовки: 0.2 мм

Требуемое уточнение: 20.0

-----
Квалитет: 5
Выполняется подбор операций
20.0 20.0
Операции определены верно.
Необходимое количество операций: 4.
Черновая, получистовая, чистовая, финишная.
Определение операций завершено.
-----

```

Рис. 5. Результат работы алгоритма

На рисунке 5 не показаны допуски для операций, но тот факт, что операции были подобраны, говорит о том, что значения допусков уже хранятся в памяти компьютера.

После создания алгоритма автоматического определения операций и назначения допусков появилась возможность реализовать расчет припуска для каждой операции по формулам [5].

После выполнения представленных алгоритмов программа выполняет уже созданный ранее алгоритм для определения шероховатости Rz и глубины дефектного слоя h (рис. 6).

```

ЭТОТ ФАЙЛ.py - C:\Users\Nekkl\Desktop\PC\PythonProject\ЭТОТ ФАЙЛ.py (3.8.3)
File Edit Format Run Options Window Help
19 Zagotovka = str(input("Укажите заготовку (прокат, поковка, штамповка, литьё): "))
20 Rz = float
21 h = float
22 if Zagotovka=="прокат":
23     type_of_prokat = str(input("Укажите точность проката (высокая, повышенная, обычная): "))
24     diametr = float(input("Укажите диаметр проката (мм): "))
25     L = float(input("Укажите длину заготовки в мм: "))
26     if type_of_prokat=="высокая":
27         if diametr<=30:
28             print("Rz =",sheet['D6'].value,"мкм;", "h =",sheet['E6'].value, "мкм.")
29             Rz = sheet['D6'].value
30             h = sheet['E6'].value
31         elif 30<diametr<=80:
32             print("Rz =",sheet['D7'].value,"мкм;", "h =",sheet['E7'].value, "мкм.")
33             Rz = sheet['D7'].value
34             h = sheet['E7'].value
35         elif 80<diametr<=180:
36             print("Rz =",sheet['D8'].value,"мкм;", "h =",sheet['E8'].value, "мкм.")
37             Rz = sheet['D8'].value
38             h = sheet['E8'].value
39         elif 180<diametr<=250:
40             print("Rz =",sheet['D9'].value,"мкм;", "h =",sheet['E9'].value, "мкм.")
41             Rz = sheet['D9'].value
42             h = sheet['E9'].value
43         else:
44             print("Введены неверные данные!")
45
46     elif type_of_prokat=="повышенная":
47         if diametr<=30:
48             print("Rz =",sheet['F6'].value,"мкм;", "h =",sheet['G6'].value, "мкм.")
49             Rz = sheet['F6'].value
50             h = sheet['G6'].value
51         elif 30<diametr<=80:
52             print("Rz =",sheet['F7'].value,"мкм;", "h =",sheet['G7'].value, "мкм.")
53             Rz = sheet['F7'].value
54             h = sheet['G7'].value
55         elif 80<diametr<=180:
56             print("Rz =",sheet['F8'].value,"мкм;", "h =",sheet['G8'].value, "мкм.")
57             Rz = sheet['F8'].value
58             h = sheet['G8'].value
59         elif 180<diametr<=250:
60             print("Rz =",sheet['F9'].value,"мкм;", "h =",sheet['G9'].value, "мкм.")
61             Rz = sheet['F9'].value
62             h = sheet['G9'].value
63         else:
64             print("Введены неверные данные!")
65
66     elif type_of_prokat=="обычная":
67         if diametr<=30:
68             print("Rz =",sheet['H6'].value,"мкм;", "h =",sheet['I6'].value, "мкм.")
69             Rz = sheet['H6'].value
70             h = sheet['I6'].value
71         elif 30<diametr<=80:
72             print("Rz =",sheet['H7'].value,"мкм;", "h =",sheet['I7'].value, "мкм.")
73             Rz = sheet['H7'].value
74             h = sheet['I7'].value
75         elif 80<diametr<=180:
76             print("Rz =",sheet['H8'].value,"мкм;", "h =",sheet['I8'].value, "мкм.")
77             Rz = sheet['H8'].value
78             h = sheet['I8'].value
79         elif 180<diametr<=250:
80             print("Rz =",sheet['H9'].value,"мкм;", "h =",sheet['I9'].value, "мкм.")
81             Rz = sheet['H9'].value
82             h = sheet['I9'].value
83         else:
84             print("Введены неверные данные!")

```

Рис. 6. Определение шероховатости и глубины дефектного слоя заготовки

Но, прежде чем перейти к определению припуска, необходимо уточнить, как заготовка закреплена на станке. Это важно, так как в системе «станок – заготовка – инструмент» возникает много погрешностей, связанных со станком, установленной в нем оснасткой, в которой закреплена заготовка, соответственно возникают погрешности закрепления, а также есть погрешности геометрической формы самой заготовки, и все эти погрешности в итоге оказывают влияние на получаемый на станке размер. Поэтому, когда допуск большой, влияние этих погрешностей будет незначительным, но, когда на станке необходимо обработать размер с точностью до микрометра, эти погрешности необходимо учесть. Числовые значения данных погрешностей известны и представлены в виде таблиц в [5], для использования их в Python они были сохранены в формате .xlsx.

```

1501 #определение пространственных отклонений ( $\Delta$ ) и погрешности установки ( $\varepsilon$ )
1502
1503 if Zagotovka=="прокат":
1504
1505     baza = str(input("укажите, как закреплена заготовка:\n"
1506                     "[1]В патроне, без поджатия задним центром\n"
1507                     "[2]В центрах\n"))
1508
1509     Dk = float
1510     Delta = float
1511     if type_of_prokat == "высокая":
1512         if L < 120:
1513             Dk = sheet['P31'].value
1514             print("Дк = ", sheet['P31'].value, "МКМ.")
1515         elif 120<L<180:
1516             Dk = sheet['Q31'].value
1517             print("Дк = ", sheet['Q31'].value, "МКМ.")
1518         elif 180<L<315:
1519             Dk = sheet['R31'].value
1520             print("Дк = ", sheet['R31'].value, "МКМ.")
1521         elif 315<L<400:
1522             Dk = sheet['S31'].value
1523             print("Дк = ", sheet['S31'].value, "МКМ.")
1524         elif 400<L<500:
1525             Dk = sheet['T31'].value
1526             print("Дк = ", sheet['T31'].value, "МКМ.")
1527     elif type_of_prokat == "повышенная":
1528         if L<120:
1529             Dk = sheet['P30'].value
1530             print("Дк = ", sheet['P30'].value, "МКМ.")
1531         elif 120<L<180:
1532             Dk = sheet['Q30'].value
1533             print("Дк = ", sheet['Q30'].value, "МКМ.")
1534         elif 180<L<315:
1535             Dk = sheet['R30'].value
1536             print("Дк = ", sheet['R30'].value, "МКМ.")
1537         elif 315<L<400:
1538             Dk = sheet['S30'].value
1539             print("Дк = ", sheet['S30'].value, "МКМ.")
1540         elif 400<L<500:
1541             Dk = sheet['T30'].value
1542             print("Дк = ", sheet['T30'].value, "МКМ.")
1543     elif type_of_prokat == "обычная":
1544         if L<120:
1545             Dk = sheet['P29'].value
1546             print("Дк = ", sheet['P29'].value, "МКМ.")
1547         elif 120<L<180:
1548             Dk = sheet['Q29'].value
1549             print("Дк = ", sheet['Q29'].value, "МКМ.")
1550         elif 180<L<315:
1551             Dk = sheet['R29'].value
1552             print("Дк = ", sheet['R29'].value, "МКМ.")
1553         elif 315<L<400:
1554             Dk = sheet['S29'].value
1555             print("Дк = ", sheet['S29'].value, "МКМ.")
1556         elif 400<L<500:
1557             Dk = sheet['T29'].value
1558             print("Дк = ", sheet['T29'].value, "МКМ.")
1559     if baza=="1":
1560         Delta = float = Dk*L
1561          $\varepsilon$  = 0
1562         print("Δ = ",Delta, "МКМ.")
1563         print("ε = ", $\varepsilon$ , "МКМ.")
1564     elif baza=="2":
1565          $\varepsilon$  = 0
1566         Dkor = float = Dk*L
1567         Dc = float = 0.25*(((Vz-Nz)**2)+1)**2
1568         Delta = float = (((Dkor)**2)+((Dc)**2))**0.5
1569         print("Δ = ",Delta, "МКМ.")
1570         print("ε = ", $\varepsilon$ , "МКМ.")
1571

```

Рис. 7. Определение пространственных отклонений и погрешности поверхностей

На рисунке 7 представлен фрагмент алгоритма для определения отклонений. В качестве заготовки используется прокат, который может быть закреплен в патроне или в центрах. Затем, в зависимости от длины заготовки и точности проката, определяется параметр Δ_k – кривизна профиля сортового проката. В зависимости от способа закрепления заготовки, при необходимости, определяются другие отклонения: $\Delta_{кор}$ – возможное коробление заготовки; Δ_c – ее смещение в закрепляемом приспособлении станка. Так как при использовании центров или трехкулачковых патронов погрешность базирования равна нулю [4, см. табл. 2.34], что также отражено в строках 1560 и 1564, то суммарная погрешность геометрической формы определяется в строках 1559 и 1567 как параметр Delta. По похожей схеме определяются погрешности для других операций и способов закрепления заготовки.

Заключение

Разработан алгоритм назначения допуска, реализация которого в среде программирования Python позволяет выполнять расчеты для всех технологических операций токарной обработки деталей и подбирать их количество, то есть позволяет выбрать от одной до четырех операций, назначить допуск для каждой операции и сразу выполнить проверку соответствия заданной точности размеров детали. Представлен алгоритм для определения отклонений на примере токарной обработки заготовки в виде проката, который может быть закреплен в патроне или в центрах. После этого определение всех необходимых параметров для расчета припуска завершено. Вторым этапом разработки программы будет написание алгоритма, реализуемого в среде программирования Python, для расчета самого припуска.

Список литературы

1. Бондаренко, Н. В. Разработка алгоритма расчета припуска на механическую обработку деталей для среды программирования Python / Н. В. Бондаренко, М. В. Соколов // Вестн. Тамб. гос. техн. ун-та. – 2022. – Т. 28, № 4. – С. 674 – 684. doi: 10.17277/vestnik.2022.04.pp.674-684
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. 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.

Modernization of the Algorithm for Calculating 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,
nikbon.2015@yandex.ru, TSTU, Tambov, Russia*

Keywords: algorithm; admission; interoperational dimensions; allowance for machining parts.

Abstract: The refined problem of calculating a rational allowance using the Python programming environment has been solved, the algorithm and program for calculating the rational allowance and determining the required number of transitions to achieve the parameters of surface roughness and dimensional accuracy specified in the design documentation have been modernized. A description is given of the developed algorithm for automatically determining operations and assigning tolerances, the implementation of which in the Python programming environment allows you to perform calculations for all turning operations and select their quantity, which significantly simplifies the work of technologists at the stage of technological preparation of machine-building production.

References

1. Bondarenko N.V., Sokolov M.V. [Development of an algorithm for calculating allowance for machining parts for the Python programming environment], *Transactions of the Tambov State Technical University*, 2022, vol. 28, no. 4, pp. 674-684. doi: 10.17277/vestnik.2022.04.pp.674-684 (In Russ., abstract in Eng.)
2. Pestretsov S.I., Altunin K.A., Sokolov M.V., Odnol'ko V.G. *Kontsepsiya sozdaniya sistemy avtomatizirovannogo proyektirovaniya protsessov rezaniya v tekhnologii mashinostroyeniya* [The concept of creating a system for automated design of cutting processes in mechanical 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
4. Altunin K.A., Sokolov M.V. [Application of neural networks for modeling the turning process], *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. Radkevich Ya.M., Timiryazev V.A., Skhirtladze A.G., Ostrovskiy M.S.; Timiryazev V.A. (Ed.). *Raschet pripuskov i mezhperekhodnykh razmerov v mashinostroyenii* [Calculation of allowances and inter-transition dimensions in mechanical engineering], Ed. 2nd, erased, Moscow: Vyssh. shkola, 2007, 272 p. (In Russ.)

Modernisierung des Algorithmus zur Berechnung der Zugaben für die mechanische Bearbeitung von Teilen für die Python-Programmierung

Zusammenfassung: Das verfeinerte Problem der Berechnung des rationalen Zuschlags ist mithilfe der Python-Programmierung gelöst, modernisiert sind der Algorithmus und das Programm zur Berechnung rationaler Aufmaße und zur

Bestimmung der erforderlichen Anzahl von Übergängen, um die in der Konstruktionsdokumentation angegebenen Parameter für Oberflächenrauheit und Maßgenauigkeit zu erreichen. Es ist der entwickelte Algorithmus zur automatischen Bestimmung von Vorgängen und Zuweisung von Toleranzen beschrieben, dessen Implementierung in der Python-Programmierung es ermöglicht, Berechnungen für alle Drehvorgänge durchzuführen und deren Anzahl auszuwählen, was die Arbeit von Technologen in der Phase der technologischen Vorbereitung der Maschinenbauproduktion erheblich vereinfacht.

Modernisation de l'algorithme de calcul de l'allocation pour le traitement mécanique des pièces pour l'environnement de programmation python

Résumé: Est résolu le problème raffiné du calcul de l'allocation rationnelle en utilisant l'environnement de programmation Python; sont modernisés l'algorithme et le programme permettant de calculer une tolérance rationnelle et de déterminer le nombre requis des transitions pour atteindre les paramètres de rugosité de surface et de précision dimensionnelle spécifiés dans la documentation de conception. Est donnée la description de l'algorithme élaboré pour déterminer automatiquement les opérations et attribuer des tolérances, dont la mise en œuvre dans l'environnement de programmation Python permet d'effectuer des calculs pour toutes les opérations de tournage et de sélectionner leur nombre, ce qui simplifie considérablement le travail des technologues au stade de préparation technologique de la production de construction de machines.

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