Using an Electronic Computer for Calculating the Performance of an Experimental Hammer Mill

Stefan Shcherbakov*, Marina* Tumanova[*]*, Elena* Kotelevskaya, *and Fedor* Simonov

KubSAU, Krasnodar, Russia

Abstract. The article discusses the need for automation in agriculture and specifically in the process of preparing feed by grinding corn grain using hammer mills. The aim of the work is to develop a Python program to calculate the productivity of an experimental hammer mill. The article also describes the experiment method for determining the degree of grinding of the grain and investigates the factors affecting the crushing process. The ultimate goal is to increase labor productivity, improve product quality, and eliminate dangerous working conditions.

1 Introduction

One of the decisive factors in increasing labor productivity in the agricultural sector is the automation of the technological process of grinding feed, as it is the most energy-intensive process. As a result, extensive research is being conducted to create automation and robotics systems in plant and animal farming. In designing livestock complexes, accurately and quickly calculating the productivity and power of the object is essential, along with selecting a complex of machines and equipment involved in the technological process, such as the process of preparing feed in feed mills.

The working hypothesis is that automating the calculation of an experimental hammer mill's productivity using a computer will increase the accuracy of calculations while reducing labor costs during the design stage.

The aim of the work is to develop a program in the Python programming language in Visual Studio for calculating the productivity of an experimental hammer mill.

The goal of automation is to increase labor productivity and efficiency, improve product quality, and eliminate humans from working in conditions that are dangerous to health. Further development of agricultural machinery will be characterized by more intensive use of information technology, automation, and the use of robotic complexes. Classification by type of some technological processes is given in Table 1.

[*] Corresponding author: tumanova-kgau@mail.ru

Table 1. Classification by the type of some technological processes.

Type of Impact	Name of Technological Operation	Name of Technological Equipment
Mechanical	Grinding	Crushers, mills
	Mixing	Mixers, feeders, dosing equipment
	Feeding	Feed distributors

Corn grain is one of the types of feed used to feed cattle, pigs, and chickens. Corn is rich in fiber, vitamins A, PP, E, and B group, minerals, and amino acids. To prepare the feed for feeding, it is crushed. For better digestion by the animal's digestive tract, grinding increases the surface area of impact, which is particularly important for feeding such feed as corn grain. The animal's dental system is poorly adapted to digesting undigested grains. Mechanical grinding can be carried out by crushing or cutting. All modern machines have all methods of grain crushing:

The technological process of crushing feed is as follows. After the electric motor is turned on and the crusher is running at a steady state, the shutter is opened. The crushed raw material from the loading hopper falls into the crushing chamber under the action of gravity. After the particle of feed is destroyed, it is removed through the holes in the sieve.

The main indicators characterizing the hammer mill and its working process are productivity, material crushing, and the energy consumption of the crushing process. These indicators are affected by factors such as the physical and chemical properties of the grain, the type of crushing device, the diameter of the sieve, the speed of the hammers, and others.

2 Experiment method

A portion of 4-5 kg of grain is crushed on a machine with a feed intensity corresponding to the average productivity of the machine. The feed is regulated by the degree of opening of the shutter. After crushing, the degree of grinding of the experimental portion of the grain is determined using a mesh classifier. The duration of the experiment is determined by a stopwatch. In the course of the work, the effect of the deck and sieve on the degree of grinding of the raw material is investigated, and the number of impacts per revolution of the drum is determined.

For processing the experimental data: the established sieve has a hole diameter of 6 mm. The number of hammers on the rotor is 24 pieces, the drum speed is 1600 rpm, and the number of riffles on the decks is 48. The weight of 1000 corn grains is 302 g, and the volume of displaced water is 300 cm3.

To calculate the productivity of the hammer mill, we will calculate:

1. The circumferential speed of the hammer movement is determined by the formula:

$$Vc = \frac{\pi^2 Do}{60} \tag{1}$$

2. Speed destruction grain determine by formula:

$$Vp = \frac{C\sqrt{Vc}}{E} \tag{2}$$

3. The grinding modulus is determined by the formula:

$$M = \frac{d1P1 + d2P2 + d3P3 + d4P4}{100} \tag{3}$$

$$V2 = \frac{\pi M^3}{6} \tag{4}$$

$$Z = \frac{V}{V2} \tag{5}$$

4. The number of hammer strikes on particles per minute is determined by the formula:

$$Zn = Km(1 + Kr + 1)n, \tag{6}$$

5. The amount of grains crushed per minute is determined by the formula:

$$N\text{з} = \frac{Zn}{Z} \tag{7}$$

6. Then the crusher productivity is determined by the formula:

$$Q = \frac{60Zn*q}{1000} \tag{8}$$

3 Flowchart of the program for calculating the productivity of an experimental hammer mill

Separating the technological process into technological operations allows us to identify its duration, sequence, cyclicality, that is, to algorithms the technological process.

A program algorithm is a sequence of actions that describes how the program should work and how to solve a specific task. It is a set of commands executed by the computer to achieve the desired result.

The program algorithm describes all the steps necessary to create a program solution and should be written in such a way as to allow the program developer to understand how the code will work.

A program algorithm can be described at different levels of complexity and detail depending on the task to be solved. It can consist of basic commands such as iterations, conditional operators, and loops, or it can contain more complex functions and technologies.

Defining a program algorithm is an important stage in the process of software development. A good algorithm should be easily readable, understandable, and efficient for implementation as program code. It should describe the principles, logic, and algorithms underlying the programmable system or application.

In addition, the approach of separating the production process into operations allows for higher accuracy in estimating the time and costs required to perform each operation. This can help improve production flow planning, optimize equipment and resource usage, and reduce the possibility of errors and rework. As for algorithms, they play a key role in software development. Although algorithms can be used to solve other tasks, such as mathematical or logical ones, they are most widely used in programming.

At the beginning of the program, the user is prompted to enter the following initial data (Table 2).

Table 2. Numerical values of the initial data.

Number of hammers (Km), pcs.	Number of riffles on decks (Kr), pcs.	Number of rotations (n), rpm	Diameter of sieve holes (Do), mm	Grinding module (M)	Weight of 1000 grains (q), g	Velocity (V), m/s	Performance (Q), kg/h
24	48	1600	6	2.45	302	300	144
28	44	1600	6	2.45	302	300	155
30	50	2000	6	2.45	200	400	167

In the body of the program (Python programming language), mathematical calculations are performed according to the calculation method presented above. As a result, the program calculates the productivity of the crusher (in kilograms per hour). The flowchart is shown in Figure 1.

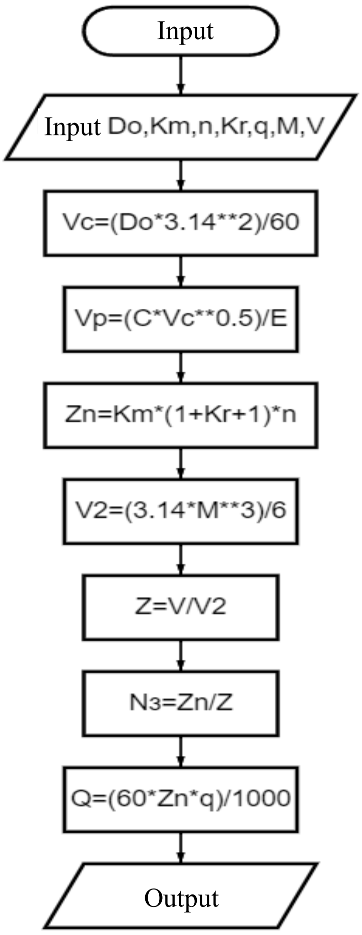


Fig. 1. Flowchart of the program.

During the scientific research work, it is possible to automatically calculate the productivity of the crusher by changing the parameters such as the number of hammers (Km), the number of revolutions (n), and the number of riffles on the decks (Kr) using the presented program. Then, the correctness of the theoretical calculation can be experimentally confirmed. Table 3 shows the changes in the parameters of the input data.

Table 3. Numerical values of the initial data.

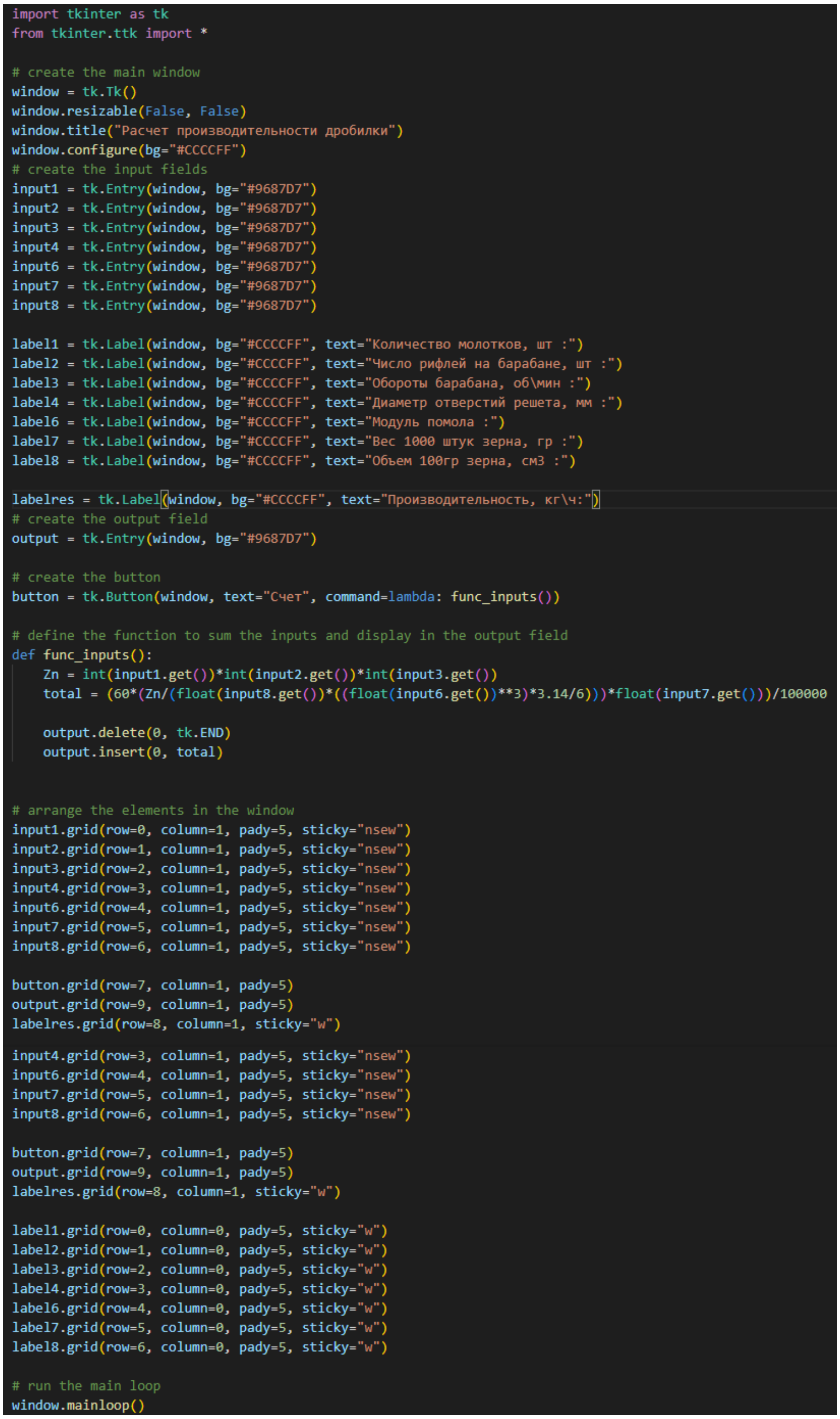
№	Number of hammers (Km), pcs.	Number of riffles on decks (Kr), pcs.	Number of rotations (n), rpm	Diameter of sieve holes (Do), mm	Grinding module (M)	Weight of 1000 grains (q), g	Velocity (V), m/s	Performance (Q), kg/h
1	24	48	1600	6	2.45	302	300	144
2	28	44	1600	6	2.45	302	300	155
3	30	50	2000	6	2.45	200	400	167
4	35	55	2500	8	3.75	350	350	105
5	40	70	3600	7	3.5	400	350	308
6	40	70	3600	5	1.5	450	200	7704

As a result of the program execution, numerical values of the productivity of the hammer mill have been obtained. The program listing is shown in Figure 2.

4 Conclusion

The use of various programming languages, including Python, is finding increasing application in agriculture for the automation of various technological processes. The automation of the process reduces costs at the stage of object design, improves calculation accuracy, allows optimization of business processes, and increases labor productivity.

During the research work, the set tasks were solved: a flowchart for calculating the productivity of a hammer mill was performed, and a program for calculating the productivity of a hammer mill was developed using Python.



```
import tkinter as tk
from tkinter.ttk import *

# create the main window
window = tk.Tk()
window.resizable(False, False)
window.title("Расчет производительности дробилки")
window.configure(bg="#CCCCFF")
# create the input fields
input1 = tk.Entry(window, bg="#9687D7")
input2 = tk.Entry(window, bg="#9687D7")
input3 = tk.Entry(window, bg="#9687D7")
input4 = tk.Entry(window, bg="#9687D7")
input6 = tk.Entry(window, bg="#9687D7")
input7 = tk.Entry(window, bg="#9687D7")
input8 = tk.Entry(window, bg="#9687D7")

label1 = tk.Label(window, bg="#CCCCFF", text="Количество молотков, шт :")
label2 = tk.Label(window, bg="#CCCCFF", text="Число рифлей на барабане, шт :")
label3 = tk.Label(window, bg="#CCCCFF", text="Обороты барабана, об\мин :")
label4 = tk.Label(window, bg="#CCCCFF", text="Диаметр отверстий решета, мм :")
label6 = tk.Label(window, bg="#CCCCFF", text="Модуль помола :")
label7 = tk.Label(window, bg="#CCCCFF", text="Вес 1000 штук зерна, гр :")
label8 = tk.Label(window, bg="#CCCCFF", text="Обьем 100гр зерна, см3 :")

labelres = tk.Label(window, bg="#CCCCFF", text="Производительность, кг\ч:")
# create the output field
output = tk.Entry(window, bg="#9687D7")

# create the button
button = tk.Button(window, text="Счет", command=lambda: func_inputs())

# define the function to sum the inputs and display in the output field
def func_inputs():
    Zn = int(input1.get())*int(input2.get())*int(input3.get())
    total = (60*(Zn/(float(input8.get())*((float(input6.get())**3)*3.14/6)))*float(input7.get()))/100000

    output.delete(0, tk.END)
    output.insert(0, total)


# arrange the elements in the window
input1.grid(row=0, column=1, pady=5, sticky="nsew")
input2.grid(row=1, column=1, pady=5, sticky="nsew")
input3.grid(row=2, column=1, pady=5, sticky="nsew")
input4.grid(row=3, column=1, pady=5, sticky="nsew")
input6.grid(row=4, column=1, pady=5, sticky="nsew")
input7.grid(row=5, column=1, pady=5, sticky="nsew")
input8.grid(row=6, column=1, pady=5, sticky="nsew")

button.grid(row=7, column=1, pady=5)
output.grid(row=9, column=1, pady=5)
labelres.grid(row=8, column=1, sticky="w")

input4.grid(row=3, column=1, pady=5, sticky="nsew")
input6.grid(row=4, column=1, pady=5, sticky="nsew")
input7.grid(row=5, column=1, pady=5, sticky="nsew")
input8.grid(row=6, column=1, pady=5, sticky="nsew")

button.grid(row=7, column=1, pady=5)
output.grid(row=9, column=1, pady=5)
labelres.grid(row=8, column=1, sticky="w")

label1.grid(row=0, column=0, pady=5, sticky="w")
label2.grid(row=1, column=0, pady=5, sticky="w")
label3.grid(row=2, column=0, pady=5, sticky="w")
label4.grid(row=3, column=0, pady=5, sticky="w")
label6.grid(row=4, column=0, pady=5, sticky="w")
label7.grid(row=5, column=0, pady=5, sticky="w")
label8.grid(row=6, column=0, pady=5, sticky="w")

# run the main loop
window.mainloop()
```

Fig. 2. The program listing

References

1. V.P. Kovalenko, V.M. Proshchak, *Mechanization of Livestock Farming*, p. 92
2. E.S Yakubovskaya, E.S Volkova, Automation of Technological Processes: Educational and Methodological Manual. (BSAU, Minsk, 2012)
3. D.M. Beazley, Python Essential Reference (Addison-Wesley Professional, Upper Saddle River, 2009)