An Investigation on the Applications of Computer-Assisted Programming in Selected Genetic Algorithms

CAN AYTAÇ[1], HÜSEYIN IRMAK[2]
[1]Nişantaşı University,
Institute of Graduate Studies,
Department of Computer Engineering,
34481742, Sarıyer-İstanbul,
TURKEY

[2]Nişantaşı University
Faculty of Engineering And Architecture,
Department of Computer Engineering
34481742, Sarıyer-İstanbul,
TURKEY

Abstract: - Mendelian genetics provides a fundamental theoretical framework for understanding the mechanisms of inheritance. However, manually calculating the possible genotype and phenotype ratios in crosses involving multiple genes and sex-linked traits is not an easy task. In this study, a Java-based genetic program was developed to systematically calculate the possible offspring genotypes and phenotypes using the genotype and sex information of two individuals. The program generates gamete combinations by considering user-defined gene–phenotype relationships, produces all possible offspring genotypes, and determines phenotypes according to dominant–recessive allele interactions. The resulting data are presented in tabular form, with genotype and phenotype ratios expressed as percentages. The developed program enables rapid and accurate analysis of Mendelian genetics–based crosses and provides evidence of its potential as an effective teaching and analytical tool in biology and genetics education.

Key-Words: - Computer programming, Java programming, Data analysis, Mathematical computing and statistics, Computational biology, Mendelian genetics, genotype and phenotype.

Received: March 16, 2026. Revised: April 11, 2026. Accepted: May 28, 2026. Published: September 7, 2026.

1 Introduction, Motivation and Literature Review

As is well known, computers and related technologies are an indispensable part of life, science, and technology. In particular, computer programming or its comprehensive applications play undeniable roles in modern science and technology by enabling complex simulations, AI-powered data analysis, and automation in various fields such as health, engineering, and space exploration. They are also indispensable for various modeling applications, big data analysis, the development of autonomous systems, and optimizing logistical, financial, or engineering processes in various ways to increase efficiency and encourage innovation.

In general, we can present some fundamental applications of computer programming in science and technology as follows:

1. In the field of scientific simulation and modeling, it involves defining complex simulations to analyze given or obtained data, simulate physical events, and model biological functions.
2. n the field of data analysis and artificial intelligence, it involves analyzing various datasets given or acquired in various fields such as medicine and finance, creating possible models, and developing or using machine learning and deep learning algorithms for pattern recognition.
3. From an engineering and robotics perspective, the goal is to systematically control, operate, and design robotic systems, automatically functioning machines, and production processes for various purposes.
4. From a health technology perspective, the aim is to create or develop diagnostic tools, necessary medical software, and advanced imaging systems to meet various needs.

5. From a space research perspective, the goal is to develop algorithms and software necessary for navigation, data collection, and control devices by guiding planned or created probes via satellites.
6. From a software and application development perspective, the aim is to create web, mobile, and system-level applications necessary for the foundation of modern digital infrastructure.
7. Within the scope of automation, the goal is to optimize workflows and replace manual tasks with artificial intelligence to minimize errors in production and logistics.
8. In quantum computing, the aim is to program next-generation computing devices to solve various potential problems beyond the limits of classical computing.
9. From a communications perspective, the aim is to provide the necessary infrastructure for advanced communication networks.

The application or use of the technologies highlighted above is, of course, quite broad in various aspects. In this respect, the aim of this particular study is to focus on the use of computers or related programs in terms of genetic structure, which is directly related to all living things. One may refer to more information given in [1], [2], [3], [4], [5], [6], [7], [8], [9], [10].

Now, let us first address some necessary information about these particular biological topics.

As we known, the discipline of genetics, which studies how hereditary traits are passed down through generations in all living organisms, is just one of the most active scientific fields in biology. In particular, Gregor Mendel's initial research on peas laid the groundwork for both mathematical and probabilistic foundations of heredity, forming the basis of modern genetics. That is, Mendelian genetics explains how phenotypes arise based on the dominance and recessiveness relationships of alleles.

In the early stages of genetics education, single-gene crosses are generally easily understood. However, with the increase in the number of genes, heterozygosity conditions, and the inclusion of sex-linked genes, the number of possible combinations increases rapidly. This can lead to both time loss and errors, especially in manual calculations. From the students' perspective, multi-gene crosses become an abstract and complex structure.

In recent years, significant advancements in computational biology and bioinformatics have made it possible to analyze various pieces of information (or data) that can be determined (or could be determined) through computer-aided tools, i.e., various computer programs or software, with undeniable contributions from different perspectives. In particular, genetic simulation software developed for educational purposes contributes to the concretization of abstract concepts of heredity in varioust ways.

The essential objective of this particular research note is to introduce a Java-based genetic calculator that can automatically calculate Mendelian genetics-based crosses and support multigene and sex-linked traits, focusing on Java programming for computer-based research, and to present the operation of this tool within an academic framework.

In special, it aims to emphasize that various calculations based on Mendelian genetics are a fundamental component of genetics education, and to remind readers that Punnett squares are commonly used in teaching genetic crosses in the written literature. However, this particular method quickly becomes complex and loses its practicality as the number of genes increases.

As a natural consequence of various computer-assisted research, various genetic simulation technologies or tools developed for educational purposes also allow users to visualize genotype and phenotype relationships. In particular, various studies show that computer-assisted genetic simulations increase students' conceptual understanding. However, some existing software only supports crosses involving a single gene or a limited number of genes, excluding sex-linked genes.

Various tools developed in the field of computational biology are generally research-oriented and can be too complex for educational use. Furthermore, closed-source software makes understanding algorithmic processes difficult. In this context, there is a need for a genetic calculator that supports user-defined gene-phenotype relationships, has a clearly defined algorithmic structure, and is educationally oriented.

For interested researchers, the extensive studies given in [11], [12], [13], [14], [15], [16], [17], [18] on Mendelian genetics and related topics contain quite detailed information.

2 The Possible Roles of Various Types of Computer Tools in the Analysis of Specific Genetic Structures

As the fundamental assumptions of the genetic model, the genetic calculator developed in this

particular study is based on the basic assumptions of Mendelian genetics. Each gene consists of two alleles, and the alleles exhibit either dominant or recessive traits. The phenotype is determined by considering the dominant alleles present in the individual's genotype. Complex genetic mechanisms such as epistasis, environmental influences, and polygenic inheritance are not considered in the model. The program to be considered is developed using the Java programming language, [19]. Java's object-oriented structure allows for the modeling of biological concepts such as genes, alleles, and individuals in a software environment.

The general operation of the program consists of the following steps:

∗ The user defines the genes and the corresponding phenotypes,

∗The genotype a nd sex information of two parent individuals is entered,

∗ Genotypes are parsed into two-letter gene pairs,

∗Possible gamete combinations are generated for each gene,

∗Calculating offspring genotypes by combining all gene and sex combinations.

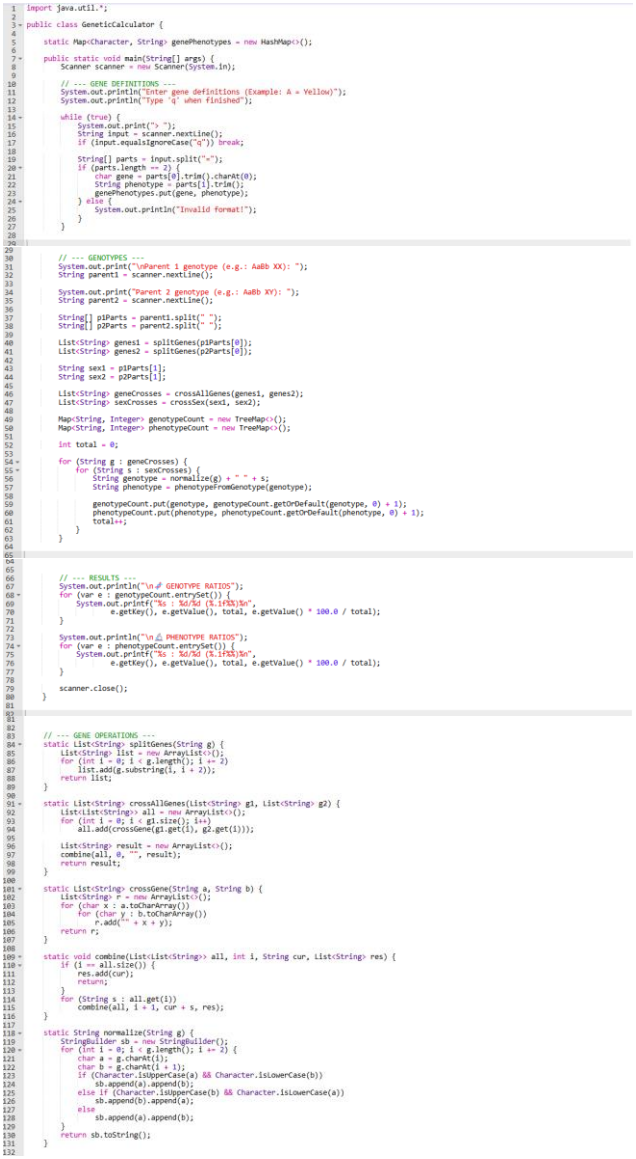
In light of the detailed information provided in the seminal work [20], we emphasize that, in terms of genotype separation and gamete formation, the input genotypes are divided into gene pairs, each consisting of two alleles. Two distinct gametes are formed for heterozygous genes, whereas a single type of gamete is produced for homozygous genes. This approach is consistent with the fundamental principles of gamete formation in Mendelian genetics.

In terms of combinatorial structure and probability calculations, the formed gametes are combined through a cross to obtain all possible offspring genotypes. The probability of each combination is expressed as a percentage by dividing it by the total number of combinations. This allows the results to be evaluated within a probabilistic framework.

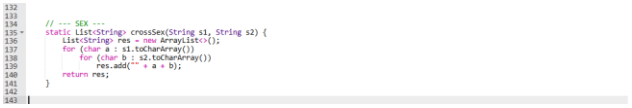
In terms of phenotype determination and integration of sex information, phenotype determination is performed for each offspring genotype. At this stage, the relevant phenotype is defined by considering the dominant alleles present in the genotype. If no dominant allele is present, the recessive phenotype is expressed. When sex-linked genes are involved, the individual's sex information is added to the phenotype definition. Thus, the analysis of hereditary traits associated with X and Y chromosomes becomes possible.

For an example application scenario, a sample crossover scenario with three genes and sex information is considered to demonstrate the program's operation. Based on the gene and phenotype information defined by the user, parental genotypes are entered, and the program calculates all possible offspring genotypes and their corresponding phenotypes.

The obtained data (or results) are also presented by Figure 1 and Figure 2, along with some output data from a long collection presented by the Java-based programming that determines genotype and phenotype ratios. The relevant outputs clearly show the percentage probability of each combination, and the table containing the program output, as shown at the end, allows interested researchers to also interpret the relevant data (results).

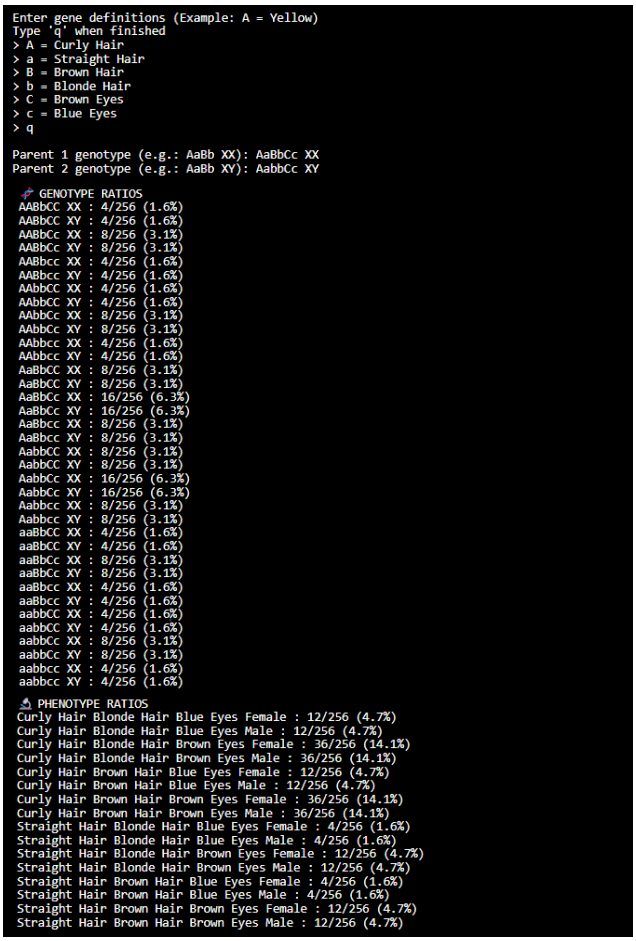


```
import java.util.*;
public class GeneticCalculator {

    static Map<Character, String> genePhenotypes = new HashMap<>();

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // --- GENE DEFINITIONS ---
        System.out.println("Enter gene definitions (Example: A = Yellow)");
        System.out.println("Type 'q' when finished");

        while (true) {
            System.out.print("> ");
            String input = scanner.nextLine();
            if (input.equalsIgnoreCase("q")) break;

            String[] parts = input.split("=");
            if (parts.length == 2) {
                char gene = parts[0].trim().charAt(0);
                String phenotype = parts[1].trim();
                genePhenotypes.put(gene, phenotype);
            } else {
                System.out.println("Invalid format!");
            }
        }


        // --- GENOTYPES ---
        System.out.print("\nParent 1 genotype (e.g.: AaBb XX): ");
        String parent1 = scanner.nextLine();

        System.out.print("Parent 2 genotype (e.g.: AaBb XY): ");
        String parent2 = scanner.nextLine();

        String[] p1Parts = parent1.split(" ");
        String[] p2Parts = parent2.split(" ");

        List<String> genes1 = splitGenes(p1Parts[0]);
        List<String> genes2 = splitGenes(p2Parts[0]);

        String sex1 = p1Parts[1];
        String sex2 = p2Parts[1];

        List<String> geneCrosses = crossAllGenes(genes1, genes2);
        List<String> sexCrosses = crossSex(sex1, sex2);

        Map<String, Integer> genotypeCount = new TreeMap<>();
        Map<String, Integer> phenotypeCount = new TreeMap<>();

        int total = 0;

        for (String g : geneCrosses) {
            for (String s : sexCrosses) {
                String genotype = normalize(g) + " " + s;
                String phenotype = phenotypeFromGenotype(genotype);

                genotypeCount.put(genotype, genotypeCount.getOrDefault(genotype, 0) + 1);
                phenotypeCount.put(phenotype, phenotypeCount.getOrDefault(phenotype, 0) + 1);
                total++;
            }
        }


        // --- RESULTS ---
        System.out.println("\n🧬 GENOTYPE RATIOS");
        for (var e : genotypeCount.entrySet()) {
            System.out.printf("%s : %d/%d (%.1f%%)%n",
                    e.getKey(), e.getValue(), total, e.getValue() * 100.0 / total);
        }

        System.out.println("\n🎨 PHENOTYPE RATIOS");
        for (var e : phenotypeCount.entrySet()) {
            System.out.printf("%s : %d/%d (%.1f%%)%n",
                    e.getKey(), e.getValue(), total, e.getValue() * 100.0 / total);
        }

        scanner.close();
    }


    // --- GENE OPERATIONS ---
    static List<String> splitGenes(String g) {
        List<String> list = new ArrayList<>();
        for (int i = 0; i < g.length(); i += 2)
            list.add(g.substring(i, i + 2));
        return list;
    }

    static List<String> crossAllGenes(List<String> g1, List<String> g2) {
        List<List<String>> all = new ArrayList<>();
        for (int i = 0; i < g1.size(); i++)
            all.add(crossGene(g1.get(i), g2.get(i)));

        List<String> result = new ArrayList<>();
        combine(all, 0, "", result);
        return result;
    }

    static List<String> crossGene(String a, String b) {
        List<String> r = new ArrayList<>();
        for (char x : a.toCharArray())
            for (char y : b.toCharArray())
                r.add("" + x + y);
        return r;
    }

    static void combine(List<List<String>> all, int i, String cur, List<String> res) {
        if (i == all.size()) {
            res.add(cur);
            return;
        }
        for (String s : all.get(i))
            combine(all, i + 1, cur + s, res);
    }

    static String normalize(String g) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < g.length(); i += 2) {
            char a = g.charAt(i);
            char b = g.charAt(i + 1);
            if (Character.isUpperCase(a) && Character.isLowerCase(b))
                sb.append(a).append(b);
            else if (Character.isUpperCase(b) && Character.isLowerCase(a))
                sb.append(b).append(a);
            else
                sb.append(a).append(b);
        }
        return sb.toString();
    }
```



```
    // --- SEX ---
    static List<String> crossSex(String s1, String s2) {
        List<String> res = new ArrayList<>();
        for (char a : s1.toCharArray())
            for (char b : s2.toCharArray())
                res.add("" + a + b);
        return res;
    }
```

Fig. 1: A Collection of Java Programs Related to the Genetic Calculator
Source: created by the authors



```
Enter gene definitions (Example: A = Yellow)
Type 'q' when finished
> A = Curly Hair
> a = Straight Hair
> B = Brown Hair
> b = Blonde Hair
> C = Brown Eyes
> c = Blue Eyes
> q

Parent 1 genotype (e.g.: AaBb XX): AaBbCc XX
Parent 2 genotype (e.g.: AaBb XY): AabbCc XY

 GENOTYPE RATIOS
AABbCC XX : 4/256 (1.6%)
AABbCC XY : 4/256 (1.6%)
AABbCc XX : 8/256 (3.1%)
AABbCc XY : 8/256 (3.1%)
AABbcc XX : 4/256 (1.6%)
AABbcc XY : 4/256 (1.6%)
AAbbCC XX : 4/256 (1.6%)
AAbbCC XY : 4/256 (1.6%)
AAbbCc XX : 8/256 (3.1%)
AAbbCc XY : 8/256 (3.1%)
AAbbcc XX : 4/256 (1.6%)
AAbbcc XY : 4/256 (1.6%)
AaBbCC XX : 8/256 (3.1%)
AaBbCC XY : 8/256 (3.1%)
AaBbCc XX : 16/256 (6.3%)
AaBbCc XY : 16/256 (6.3%)
AaBbcc XX : 8/256 (3.1%)
AaBbcc XY : 8/256 (3.1%)
AabbCC XX : 8/256 (3.1%)
AabbCC XY : 8/256 (3.1%)
AabbCc XX : 16/256 (6.3%)
AabbCc XY : 16/256 (6.3%)
Aabbcc XX : 8/256 (3.1%)
Aabbcc XY : 8/256 (3.1%)
aaBbCC XX : 4/256 (1.6%)
aaBbCC XY : 4/256 (1.6%)
aaBbCc XX : 8/256 (3.1%)
aaBbCc XY : 8/256 (3.1%)
aaBbcc XX : 4/256 (1.6%)
aaBbcc XY : 4/256 (1.6%)
aabbCC XX : 4/256 (1.6%)
aabbCC XY : 4/256 (1.6%)
aabbCc XX : 8/256 (3.1%)
aabbCc XY : 8/256 (3.1%)
aabbcc XX : 4/256 (1.6%)
aabbcc XY : 4/256 (1.6%)

 PHENOTYPE RATIOS
Curly Hair Blonde Hair Blue Eyes Female : 12/256 (4.7%)
Curly Hair Blonde Hair Blue Eyes Male : 12/256 (4.7%)
Curly Hair Blonde Hair Brown Eyes Female : 36/256 (14.1%)
Curly Hair Blonde Hair Brown Eyes Male : 36/256 (14.1%)
Curly Hair Brown Hair Blue Eyes Female : 12/256 (4.7%)
Curly Hair Brown Hair Blue Eyes Male : 12/256 (4.7%)
Curly Hair Brown Hair Brown Eyes Female : 36/256 (14.1%)
Curly Hair Brown Hair Brown Eyes Male : 36/256 (14.1%)
Straight Hair Blonde Hair Blue Eyes Female : 4/256 (1.6%)
Straight Hair Blonde Hair Blue Eyes Male : 4/256 (1.6%)
Straight Hair Blonde Hair Brown Eyes Female : 12/256 (4.7%)
Straight Hair Blonde Hair Brown Eyes Male : 12/256 (4.7%)
Straight Hair Brown Hair Blue Eyes Female : 4/256 (1.6%)
Straight Hair Brown Hair Blue Eyes Male : 4/256 (1.6%)
Straight Hair Brown Hair Brown Eyes Female : 12/256 (4.7%)
Straight Hair Brown Hair Brown Eyes Male : 12/256 (4.7%)
```

Fig. 2: Sample Outputs Generated by the Java-Based Genetic Calculator
Source: created by the authors

3 Conclusion and Final Words

In light of information from computational mathematics along with statistics, this special study primarily provides information on various computer-aided applications, includes selected topics related to genetics, and presents relevant references.

An examination of the outputs presented in Figure 1 and Figure 2, generated by the Java program, indicates that the calculated genotype and phenotype ratios are consistent with classical Mendelian cross results. The systematic and clear presentation of results for multigene crosses further demonstrates the program's ease of use.

Moreover, in the analysis of sex-linked genes, integrating sex information into phenotype determination enhances the realism and instructional value of the results, thereby increasing the educational effectiveness of the program. The developed Java-based genetic calculator offers significant advantages for analyzing Mendelian genetics–based crosses. By minimizing errors commonly encountered in manual calculations, it also accelerates the analysis of complex gene combinations across various contexts.

Finally, since the program is based on classical Mendelian assumptions, it has certain limitations. Advanced genetic mechanisms, such as epistasis, environmental effects, and polygenic inheritance, were beyond the scope of this study. Readers may also consult the academic studies cited in [19], [21], [22], [23], [24], [25], [26], [27], [28], which contain more (special) information directly or indirectly related to certain types of computer programs and/or the objectives of this investigation.

References:

[1] G. David, The Science of Programming, Springer-Verlag, 1981.
[2] A. Hyman, Charles Babbage: Pioneer of the Computer, Oxford University Press, 1982.
[3] D. E. Knuth, Selected Papers on Computer Science, CSLI Publications, Cambridge University Press, 1996.
[4] W. K. Brian, The Practice of Programming, Pearson, 1999.
[5] A. Ralston, E. D. Reilly and D. Hemmendinger, Encyclopedia of Computer Science, Grove's Dictionaries, 2000.
[6] D. R. Edwin, Milestones in Computer Science and Information Technology, Greenwood Publishing Group, 2003.
[7] A. B. Tucker, Computer Science Handbook, Chapman and Hall: CRC, 2004.
[8] P. A. Laplante, Encyclopedia of Software Engineering Three-Volume Set (Print), CRC Press, 2020.
[9] R. W. P. Luk, Insight in how computer science can be a science, Science & Philosophy, Vol. 8, No. 2, 2020, pp. 17-47.
[10] R. W. Sebesta, Concepts of Programming Languages, Pearson, 2023.
[11] W Bateson and G. Mendel, Mendel's principles of heredity, Cambridge: Cambridge University Press, 1909.
[12] B. C. A. Windle, Mendel and His Theory of Heredity, A Century of Scientific Thought and Other Essays, Burns & Oates, 1915.

[13] R. C. Punnett, Mendelism, London: Macmillan, 1922.
[14] H. Iltis, Gregor Johann Mendel. Leben, Werk und Wirkung, Berlin: J. Springer, 1924.
[15] B. L. V. D. Waerden, Mendel's Experiments. Centaurus, Vol. 12, No. 4, 1968, 275-88.
[16] A. Zumkeller and H. A. Adolar, Recently Discovered Sermon Sketches of Gregor Mendel, Folia Mendeliana, Vol. 6, 1971, pp. 24-52.
[17] G. Mendel, A. F. Corcos, F. V. Monaghan and M. C. Weber, Gregor Mendel's Experiments on Plant Hybrids: A Guided Study, Rutgers University Press, 1993.
[18] V. Orel, Gregor Mendel: The first geneticist, Oxford: Oxford University Press, 1996.
[19] K. Arnold, J. Gosling and D. Holmes, The Java programming language, Addison-Wesley Professional, 2005.
[20] W. S. Klug, M. R. Cummings, C. A. Spencer and M. A. Palladino, Concepts of Genetics, Pearson, 2019.
[21] D. L. Parnas, Software engineering programmes are not computer science programmes, Annals of Software Engineering, Vol. 6, No. 1-4, 1998, pp. 19–37.
[22] W. P. Terrence and V. Z. Marvin, Programming Languages: Design and Implementation, Prentice Hall 2000.
[23] M. Tedre, The science of computing: Shaping a discipline, Taylor and Francis, CRC Press, 2014.
[24] P. J. Denning, D. E. Comer, D. Gries, M. C. Mulder, A. Tucker, A. J. Turner and P. R. Young, Computing as a discipline, Computer, Vol. 22, No. 2, 1989, pp. 63-70.
[25] B. A. Pierce, Genetics: A Conceptual Approach, W. H. Freeman, 2017.
[26] A. J. F. Griffiths, S. R. Wessler, R. C. Lewontin and S. B. Carroll, Introduction to Genetic Analysis, W. H. Freeman, 2015.
[27] M. R. Cummings, Bioinformatics and Computational Genetics: Modeling Genotypes and Phenotypes in Java, Academic Press, 2015.
[28] J. Smith and L. Johnson, Mendelian Inheritance Simulations Using Java, Journal of Biological Education, Vol. 52, No. 3, 2018, pp. 234-242.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors contributed to the present research, at all stages from the formulation of the problem to the final findings and solutions.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This research received no external funding.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.