**PROJECT**

**OF**

**COMPUTER SCIENCE ON**

**JAVA**

**Submitted By:**

**Divya Gupta**

**Shiwalik International School**

**ACKNOWLEDGEMENT**

I WOULD LIKE TO EXPRESS MY SPECIAL THANKS OF GRATITUDE TO MY TEACHER, MRS. SHARANKHLA PANDEY , WHO GAVE ME THE OPPORTUNITY TO DO THIS PROJECT OF COMPUTER SCIENCE, WHICH HELPED ME IN LEARNING MANY NEW AND USEFUL THINGS.

I WOULD ALSO LIKE TO THANK MY PARENTS, SISTERS AND FRIENDS WHO HELPED ME IN FINALISING THIS PROJECT WITHIN THE GIVEN TIME PERIOD.

DIVYA GUPTA

12TH – SCIENCE

SHIWALIK INTERNATIONAL SCHOOL

**CERTIFICATE**

THIS IS TO CERTIFY THAT DIVYA GUPTA OF CLASS 12TH SCIENCE FROM SHIWALIK INTERNATIONAL SCHOOL HAS SUCCESSFULLY CARRIED OUT HIS PROJECT ASSESSMENT OF COMPUTER SCIENCE UNDER MY GUIDANCE AND SUPERVISION.

MRS. SHARANKHLA PANDEY

SIGNATURE OF INTERNAL EXAMINER- …………………..

SIGNATURE OF EXTERNAL EXAMINER-……………………

**BIBLIOGRAPHY**

I WOULD LIKE TO CONCLUDE THIS PROJECT BY MENTIONING ALL THE SOURCES FROM WHICH I TOOK OUT THE MATERIAL NEEDED TO COMPLETE THIS PROJECT SUCCESSFULLY.

**1. www.google.com**

**2. https://en.wikipedia.org**

**3. Understanding ISC computer science for class 12th – APC Publisher**

**By V.K. Pandey and D.K. Dey**

**4. Computer Science 12th –By Sumita Arora**

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO.** | **PROGRAMS** | **PAGE NO.** | |
| 1. | To generate prime factors of a number |  |
| 2. | Smith number using Inheritance |  |
| 3. | Longest string in a sentence |  |
| 4. | Recursion |  |
| 5. | Magic number |  |
| 6. | Transpose of an array |  |
| 7. | Inheritance |  |
| 8. | String |  |
| 9. | Bank Task |  |
| 10. | Kaprekar number |  |
| 11. | Frequency of words in ascending order |  |
| 12. | Permutation and Combination |  |
| 13. | String |  |
| 14. | Perfect number |  |
| 15. | Wondrous square |  |
| 16. | Anagram in a string |  |
| 17. | Automorphic number |  |
| 18. | Special number |  |
| 19. | Coordinates |  |
| 20. | String pattern |  |
| 21. | String |  |
| 22. | Reverse of a string |  |
| 23. | String palindrome or not |  |
| 24. | Bubble Sort |  |
| 25. | Selection Sort |  |
| 26. | Stack |  |
| 27. | Queue |  |
| 28. | Scalar matrix |  |
| 29. | Diagonal matrix |  |
| 30. | Linear Search |  |
| 31. | Binary Search |  |
| 32. | Function Overloading |  |
| 33. | Twin Prime numbers |  |
| 34. | Circular Prime number |  |
| 35. | Binary search (using Recursion) |  |
| 36. | String |  |
| 37. | Fibonacci String |  |
| 38. | Merging Arrays inside object |  |
| 39. | Inheritance |  |
| 40. | Fibonacci series using Recursion |  |

**BLUEJ**

**INTRODUCTION**

**BLUEJ** is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for the [Java programming language](https://en.wikipedia.org/wiki/Java_(programming_language)), developed mainly for educational purposes, but also suitable for small-scale [software development](https://en.wikipedia.org/wiki/Software_development). It runs with the help of JDK (Java Development Kit). Java is a programming language and a platform which works on BlueJ.

BlueJ was developed to support the learning and teaching of [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming), and its design differs from other development environments as a result.[[2]](https://en.wikipedia.org/wiki/BlueJ#cite_note-Ref_-2) The main screen graphically shows the class structure of an application under development (in a [UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language)-like diagram), and objects can be interactively created and tested. This interaction facility, combined with a clean, simple user interface, allows easy experimentation with objects under development. Object-oriented concepts ([classes](https://en.wikipedia.org/wiki/Class_(computer_science)), [objects](https://en.wikipedia.org/wiki/Object_(computer_science)), communication through [method calls](https://en.wikipedia.org/wiki/Method_(computer_science))) are represented visually and in its interaction design in the interface.

**HISTORY**

The development of BlueJ was started in 1999 by [Michael Kölling](https://en.wikipedia.org/wiki/Michael_K%C3%B6lling) and [John Rosenberg](https://en.wikipedia.org/wiki/John_Rosenberg_(academic)) at [Monash University](https://en.wikipedia.org/wiki/Monash_University" \o "Monash University), as a successor to the [Blue](https://en.wikipedia.org/w/index.php?title=Blue_programming_language&action=edit&redlink=1) system. BlueJ is an IDE. Blue was an integrated system with its own programming language and environment. BlueJ implements the Blue environment design for the Java programming language.

BlueJ is currently being maintained by a team at the [University of Kent](https://en.wikipedia.org/wiki/University_of_Kent), Canterbury, England – where Kölling now lectures.

In March 2009, the BlueJ project became [free and open source software](https://en.wikipedia.org/wiki/Free_and_open_source_software), and licensed under [GNU GPL](https://en.wikipedia.org/wiki/GNU_GPL) with the [classpath exception](https://en.wikipedia.org/wiki/Classpath_exception" \l "The_classpath_exception" \o "Classpath exception).

**Features**

The features of BlueJ are designed as an aid to learning object-oriented programming concepts, as well as an aid to program development itself. As a result, some commonly available tools differ from other [environments](http://www.liquisearch.com/what_are_environments), some tools are absent, and other tools are provided not commonly found in [development environments](http://www.liquisearch.com/what_are_development_environments). The features include:

* Representation of object orientation – the concepts of classes and objects are presented in a visual manner. Both have distinct visual representations, and it is easy to demonstrate that code is associated directly with a class, that an object has a class type, and that the class type of an object directly determines its available methods and fields as well as its behavior.
* Simplicity of the interface – The user interface is simpler than in full-scale professional environments, and thus easier to learn. The goal is for the development environment to "disappear". Students should be able to concentrate on the programming task without the need to struggle with the [environment](http://www.liquisearch.com/what_is_environment).
* Object interaction – Programmers are able to experiment with objects by creating objects on the "Object Bench" and calling individual methods interactively (including passing of parameters and inspection of results). This is intended to encourage early and frequent testing, and experimentation to help understand the programming model.
* The "Code Pad" – The code pad is a tool that instantly evaluates arbitrary Java expressions and statements.
* Regression testing – BlueJ supports regression testing via an integration of JUnit. In addition to manual writing of JUnit classes, interactive tests can be recorded and JUnit test cases can be automatically created from these recordings.
* Group work support – BlueJ provides simple support for group work via a subset of CVS and Subversion functionality.
* Java ME support – Java ME (Micro Edition) projects can be developed and deployed from [BlueJ](http://www.liquisearch.com/what_is_bluej" \o "What is BlueJ?).
* Flexible extension system – Extensions (a.k.a. Plug-Ins) can be developed using a public extensions API to extend the functionality of the core environment.
* JAR files and applets – Creation of executable JAR files and applets is built into the system.
* Translations – The BlueJ system interface has been translated into at least 14 different natural languages.

The visual interaction features of BlueJ were designed to allow one to delay the introduction of certain programming concepts considered difficult or problematic by educators. These include:

* Syntax such as public static void main(String args) – which requires unnecessarily introducing the keywords public and static, as well as method arguments and arrays. BlueJ allows arbitrary classes to be instantiated and arbitrary methods to be invoked interactively.
* Program input/output. In BlueJ both the parameter values for and the return values from method calls can be entered/inspected directly, so there is no need for students to deal with terminal I/O or write graphical user interfaces when they are still struggling with the basic concepts of [programming](http://www.liquisearch.com/what_is_programming).

The program is a fully functional development tool, commonly used by schools, colleges, and universities.

Some of the features BlueJ pioneered in modern development environments, such as interactive instantiation, direct object interaction, unit test recording, the editor "navigation view", and scope highlighting, are generally useful for software [development](http://www.liquisearch.com/what_is_development), however, are not usually found in other development environments. Some attempts were made to replicate some of these [features](http://www.liquisearch.com/what_are_features): For example, Microsoft's Visual Studio replicated BlueJ's object bench feature, however, it remained somewhat obscure because it is not included in all versions of that environment.