



## **PROJECT REPORT**

**TITLE : INTERACTIVE QUIZ APPLICATION**

---

<b>Submitted By</b>	<b>: Vennapusa Tharun Sainath Reddy</b>
<b>Submitted To</b>	<b>: Global Quest Technologies</b>
<b>Duration</b>	<b>: January 2026</b>

---

# 1.ABSTRACT

- The *KBC – Java Console Based Quiz Application* is an interactive quiz game developed using core Java concepts. Inspired by the popular television show *Kaun Banega Crorepati*, this application simulates a real-time quiz environment in a console interface. The program presents a predefined set of ten multiple-choice questions to the player, each associated with a specific prize amount.
- The player advances through the game only by selecting the correct answer. Every correct response increases the prize money and promotes the player to the next question. The application includes two lifelines—**50-50** and **Audience Poll**—that can be used once throughout the entire game. The 50-50 lifeline removes two incorrect options, while the Audience Poll provides guidance by highlighting the most likely correct option.
- The project demonstrates practical implementation of core Java features such as arrays, loops, conditional statements, boolean flags, and user input handling using the Scanner class. ANSI escape codes are used to enhance the user experience by displaying colored outputs for correct answers, wrong answers, and lifeline messages. The game ends immediately upon a wrong answer, maintaining excitement and realism. This project serves as a strong foundation for understanding control flow and logic building in Java.

## 2.PROBLEM STATEMENT

- Many basic console-based quiz programs are limited to simple question-and-answer mechanisms. They often lack structured progression, prize tracking, and controlled assistance features. Such programs do not effectively simulate a real-world quiz experience.
- There is a need for a Java-based quiz application that:
  - Displays questions in a fixed and structured sequence
  - Validates user input at every stage
  - Rewards players with incremental prize money
  - Allows limited assistance through lifelines
  - Restricts lifelines to one-time use
  - Terminates the game immediately upon incorrect answers
  - Enhances interaction using formatted and coloured outputs
- The challenge is to design and implement a robust console application using only core Java concepts that offers an engaging and realistic quiz experience while maintaining simplicity and clarity in program flow.

### 3.INTRODUCTION

- Quiz applications are widely used in education and entertainment. They help in assessing knowledge, improving learning, and increasing user engagement. A console-based quiz system is particularly useful for beginners in programming, as it combines logical decision-making, iteration, and user interaction in a controlled environment.
- The KBCQuiz application is developed entirely using core Java without any external libraries or frameworks. The game begins by collecting player details and displaying rules. It then presents a sequence of ten questions, each with four options and an additional option for lifeline usage. The user must answer correctly to move forward.
- The application ensures:
  - Real-time validation of answers
  - Immediate feedback on correctness
  - Controlled use of lifelines
  - Game termination on wrong input
- By simulating a real KBC-style game in the console, the project bridges entertainment with learning and demonstrates how simple programming constructs can be used to build engaging applications.

## 4.OBJECTIVES OF PROJECT

- The primary objectives of the KBCQuiz application are:
  - To design a console-based quiz application using core Java
  - Teach and demonstrate the use of:
    - Arrays
    - Loops
    - Conditional statements
    - Boolean flags
    - User input handling
  - To present a fixed sequence of ten multiple-choice questions
  - To provide immediate feedback for each answer
  - To reward correct answers with increasing prize values
  - To implement two lifelines:
    - 50-50 – removes two wrong options
    - Audience Poll – provides guidance on the likely answer
  - To restrict each lifeline to one-time usage
  - To terminate the game instantly on wrong answers
  - To enhance the user experience with coloured console output

## 5.SYSTEM REQUIREMENTS

### ➤ **Hardware Requirements:**

- Processor: Intel Core i3 or equivalent
- RAM: Minimum 4 GB
- Storage: At least 100 MB free space
- Input Device: Keyboard
- Output Device: Monitor

### ➤ **Software Requirements:**

- Operating System: Windows / Linux / macOS
- Programming Language: Java
- Java Version: JDK 8 or above
- IDE / Editor: Eclipse, IntelliJ IDEA, VS Code, or any Java-supported editor

## 6.PROGRAM LOGIC AND FLOW

- Program execution starts from the main() method.
- A Scanner object is created for user input.
- Player name and age are collected.
- Game rules are displayed.
- A for loop iterates through all questions.
- For each question:
  - Question and four options are displayed
  - A fifth option is shown for lifeline usage
- User input is read:
  - If the user selects a normal option (1–4), the answer is validated
  - If the user selects option 5:
    - Available lifelines are shown
    - The selected lifeline is executed
- Lifeline Behaviour:
  - **50-50**: Displays only one correct and one random wrong option
  - **Audience Poll**: Suggests the correct option
- Validation:
  - Correct answer → prize updated → next question
  - Wrong answer → game ends immediately
- If all questions are answered correctly, a final congratulatory message is displayed.

## 7.SOURCE CODE

```
1 package project;
2 import java.util.Scanner;
3 public class KBCQuiz {
4     public static final String GREEN = "\u001B[32m";
5     public static final String RESET = "\u001B[0m";
6     public static final String RED = "\u001B[31m";
7     public static final String BLUE = "\u001B[34m";
8     public static void main(String[] args) {
9         Scanner sc = new Scanner(System.in);
10        System.out.println(RED + "===== " + RESET);
11        System.out.println(RED + "          WELCOME TO KBC QUIZ          " + RESET);
12        System.out.println(RED + "===== " + RESET);
13        System.out.print("Enter Player Name: ");
14        String name = sc.nextLine();
15        System.out.print("Enter Age: ");
16        int age = sc.nextInt();
17        System.out.println(GREEN + "\nWelcome " + name + " (" + age + " years)" + RESET);
18        System.out.println("-----");
19        System.out.println(" Rules:");
20        System.out.println("1. Each question has 5 options");
21        System.out.println("2. Option 5 = Lifeline");
22        System.out.println("3. Lifelines: 50-50 & Audience Poll (Once each)");
23        System.out.println("4. Wrong answer ends the game");
24        System.out.println("-----\n");
25        String[] questions = {
26            "1. Which keyword is used to inherit a class in Java?",
27            "2. Which planet has the largest number of moons?",
28            "3. What will be the output of: System.out.println(5 + 2 + \"3\");",
29            "4. Who directed the movie *Baahubali*?",
30            "5. Which data structure works on LIFO principle?",
31            "6. Which is the longest river in the world?",
32            "7. What is the time complexity of Binary Search?",
33            "8. Who won the Cricket World Cup 2011?",
34            "9. Which HTML tag is used to create a hyperlink?",
35            "10. What will be the output of: System.out.println(\"Java\".length());"
36        };
37        String[][] options = {
38            {"1) implements", "2) inherit", "3) extends", "4) super"},
39            {"1) Earth", "2) Mars", "3) Jupiter", "4) Saturn"},
40            {"1) 10", "2) 73", "3) 523", "4) 32"},
41            {"1) S. S. Rajamouli", "2) Shankar", "3) Mani Ratnam", "4) Atlee"},
42            {"1) Queue", "2) Stack", "3) Array", "4) Tree"},
43            {"1) Amazon", "2) Nile", "3) Yangtze", "4) Mississippi"},
44            {"1) O(n)", "2) O(log n)", "3) O(n^2)", "4) O(1)"},
45            {"1) Australia", "2) India", "3) England", "4) Sri Lanka"},
46            {"1) <link>", "2) <href>", "3) <a>", "4) <url>"},
47            {"1) 3", "2) 4", "3) 5", "4) 6"}
48        };
49        int[] answers = {3, 4, 2, 1, 2, 2, 2, 3, 2};
50        int[] prize = {1000, 2000, 5000, 10000, 20000, 40000, 80000, 160000, 320000, 640000};
51        boolean fiftyUsed = false;
52        boolean audienceUsed = false;
53        for (int i = 0; i < questions.length; i++) {
54            System.out.println(BLUE + "\n----- " + RESET);
55            System.out.println(questions[i]);
56            for (int j = 0; j < 4; j++) {
57                System.out.println(options[i][j]);
58            }
59            System.out.println("5) Help (Lifeline)");
60            System.out.print("Choose your option: ");
61            int choice = sc.nextInt();
62            if (choice == 5) {
63                if (fiftyUsed && audienceUsed) {
64                    System.out.println(RED + "No lifelines left!" + RESET);
65                    i--;
66                    continue;
67                }
68                System.out.println("Choose Lifeline:");
69                if (!fiftyUsed) System.out.println("1) 50-50");
70                if (!audienceUsed) System.out.println("2) Audience Poll");
71                int life = sc.nextInt();
72                if (life == 1 && !fiftyUsed) {
73                    fiftyUsed = true;
74                    System.out.println(BLUE + "50-50 Lifeline Activated!" + RESET);
75                }
76            }
77        }
78    }
79 }
```



```

74      System.out.println(BLUE + "50-50 Lifeline Activated!" + RESET);
75      boolean[] show = new boolean[4];
76      int correct = answers[i] - 1;
77      show[correct] = true;
78      int randomWrong;
79      do {
80          randomWrong = (int) (Math.random() * 4);
81      } while (randomWrong == correct);
82      show[randomWrong] = true;
83      System.out.println(BLUE + "\n" + questions[i] + RESET);
84      for (int k = 0; k < 4; k++) {
85          if (show[k]) {
86              System.out.println(options[i][k]);
87          }
88      }
89      System.out.print("Choose your option: ");
90      int newChoice = sc.nextInt();
91      if (newChoice == answers[i]) {
92          System.out.println(GREEN + "✓ Correct Answer!" + RESET);
93          System.out.println(GREEN + "🎉 Congratulations " + name +
94              ", your prize is Rs." + prize[i] + RESET);
95      } else {
96          System.out.println(RED + "✗ Wrong Answer!" + RESET);
97          System.out.println(RED + "🎮 Game Over " + name + RESET);
98          System.out.println("You won Rs." + (i > 0 ? prize[i - 1] : 0));
99          return;
100      }
101      continue;
102  }
103  else if (life == 2 && !audienceUsed) {
104      audienceUsed = true;
105      System.out.println(BLUE + "Audience Poll:" + RESET);
106      System.out.println(BLUE + "Option " + answers[i] + " has highest votes!" + RESET);
107      i--;
108      continue;
109  }
110  else {
111  }
112  }
113  }
114  }
115  }
116  }
117  }
118  }
119  }
120  }
121  }
122  }
123  }
124  }
125  }
126  }
127  }
128  }
129  }
130  }
131  }
132  }
133  }
134  }
135  }
136  }
137  }
138  }
139  }
140  }
141  }
142  }
143  }
144  }
145  }
146  }
147  }
148  }
149  }
150  }
151  }
152  }
153  }
154  }
155  }
156  }
157  }
158  }
159  }
160  }
161  }
162  }
163  }
164  }
165  }
166  }
167  }
168  }
169  }
170  }
171  }
172  }
173  }
174  }
175  }
176  }
177  }
178  }
179  }
180  }
181  }
182  }
183  }
184  }
185  }
186  }
187  }
188  }
189  }
190  }
191  }
192  }
193  }
194  }
195  }
196  }
197  }
198  }
199  }
200  }
201  }
202  }
203  }
204  }
205  }
206  }
207  }
208  }
209  }
210  }
211  }
212  }
213  }
214  }
215  }
216  }
217  }
218  }
219  }
220  }
221  }
222  }
223  }
224  }
225  }
226  }
227  }
228  }
229  }
230  }
231  }
232  }
233  }
234  }
235  }
236  }
237  }
238  }
239  }
240  }
241  }
242  }
243  }
244  }
245  }
246  }
247  }
248  }
249  }
250  }
251  }
252  }
253  }
254  }
255  }
256  }
257  }
258  }
259  }
260  }
261  }
262  }
263  }
264  }
265  }
266  }
267  }
268  }
269  }
270  }
271  }
272  }
273  }
274  }
275  }
276  }
277  }
278  }
279  }
280  }
281  }
282  }
283  }
284  }
285  }
286  }
287  }
288  }
289  }
290  }
291  }
292  }
293  }
294  }
295  }
296  }
297  }
298  }
299  }
300  }
301  }
302  }
303  }
304  }
305  }
306  }
307  }
308  }
309  }
310  }
311  }
312  }
313  }
314  }
315  }
316  }
317  }
318  }
319  }
320  }
321  }
322  }
323  }
324  }
325  }
326  }
327  }
328  }
329  }
330  }
331  }
332  }
333  }
334  }
335  }
336  }
337  }
338  }
339  }
340  }
341  }
342  }
343  }
344  }
345  }
346  }
347  }
348  }
349  }
350  }
351  }
352  }
353  }
354  }
355  }
356  }
357  }
358  }
359  }
360  }
361  }
362  }
363  }
364  }
365  }
366  }
367  }
368  }
369  }
370  }
371  }
372  }
373  }
374  }
375  }
376  }
377  }
378  }
379  }
380  }
381  }
382  }
383  }
384  }
385  }
386  }
387  }
388  }
389  }
390  }
391  }
392  }
393  }
394  }
395  }
396  }
397  }
398  }
399  }
400  }
401  }
402  }
403  }
404  }
405  }
406  }
407  }
408  }
409  }
410  }
411  }
412  }
413  }
414  }
415  }
416  }
417  }
418  }
419  }
420  }
421  }
422  }
423  }
424  }
425  }
426  }
427  }
428  }
429  }
430  }
431  }
432  }
433  }
434  }
435  }
436  }
437  }
438  }
439  }
440  }
441  }
442  }
443  }
444  }
445  }
446  }
447  }
448  }
449  }
450  }
451  }
452  }
453  }
454  }
455  }
456  }
457  }
458  }
459  }
460  }
461  }
462  }
463  }
464  }
465  }
466  }
467  }
468  }
469  }
470  }
471  }
472  }
473  }
474  }
475  }
476  }
477  }
478  }
479  }
480  }
481  }
482  }
483  }
484  }
485  }
486  }
487  }
488  }
489  }
490  }
491  }
492  }
493  }
494  }
495  }
496  }
497  }
498  }
499  }
500  }
501  }
502  }
503  }
504  }
505  }
506  }
507  }
508  }
509  }
510  }
511  }
512  }
513  }
514  }
515  }
516  }
517  }
518  }
519  }
520  }
521  }
522  }
523  }
524  }
525  }
526  }
527  }
528  }
529  }
530  }
531  }
532  }
533  }
534  }
535  }
536  }
537  }
538  }
539  }
540  }
541  }
542  }
543  }
544  }
545  }
546  }
547  }
548  }
549  }
550  }
551  }
552  }
553  }
554  }
555  }
556  }
557  }
558  }
559  }
560  }
561  }
562  }
563  }
564  }
565  }
566  }
567  }
568  }
569  }
570  }
571  }
572  }
573  }
574  }
575  }
576  }
577  }
578  }
579  }
580  }
581  }
582  }
583  }
584  }
585  }
586  }
587  }
588  }
589  }
590  }
591  }
592  }
593  }
594  }
595  }
596  }
597  }
598  }
599  }
600  }
601  }
602  }
603  }
604  }
605  }
606  }
607  }
608  }
609  }
610  }
611  }
612  }
613  }
614  }
615  }
616  }
617  }
618  }
619  }
620  }
621  }
622  }
623  }
624  }
625  }
626  }
627  }
628  }
629  }
630  }
631  }
632  }
633  }
634  }
635  }
636  }
637  }
638  }
639  }
640  }
641  }
642  }
643  }
644  }
645  }
646  }
647  }
648  }
649  }
650  }
651  }
652  }
653  }
654  }
655  }
656  }
657  }
658  }
659  }
660  }
661  }
662  }
663  }
664  }
665  }
666  }
667  }
668  }
669  }
670  }
671  }
672  }
673  }
674  }
675  }
676  }
677  }
678  }
679  }
680  }
681  }
682  }
683  }
684  }
685  }
686  }
687  }
688  }
689  }
690  }
691  }
692  }
693  }
694  }
695  }
696  }
697  }
698  }
699  }
700  }
701  }
702  }
703  }
704  }
705  }
706  }
707  }
708  }
709  }
710  }
711  }
712  }
713  }
714  }
715  }
716  }
717  }
718  }
719  }
720  }
721  }
722  }
723  }
724  }
725  }
726  }
727  }
728  }
729  }
730  }
731  }
732  }
733  }
734  }
735  }
736  }
737  }
738  }
739  }
740  }
741  }
742  }
743  }
744  }
745  }
746  }
747  }
748  }
749  }
750  }
751  }
752  }
753  }
754  }
755  }
756  }
757  }
758  }
759  }
760  }
761  }
762  }
763  }
764  }
765  }
766  }
767  }
768  }
769  }
770  }
771  }
772  }
773  }
774  }
775  }
776  }
777  }
778  }
779  }
780  }
781  }
782  }
783  }
784  }
785  }
786  }
787  }
788  }
789  }
790  }
791  }
792  }
793  }
794  }
795  }
796  }
797  }
798  }
799  }
800  }
801  }
802  }
803  }
804  }
805  }
806  }
807  }
808  }
809  }
810  }
811  }
812  }
813  }
814  }
815  }
816  }
817  }
818  }
819  }
820  }
821  }
822  }
823  }
824  }
825  }
826  }
827  }
828  }
829  }
830  }
831  }
832  }
833  }
834  }
835  }
836  }
837  }
838  }
839  }
840  }
841  }
842  }
843  }
844  }
845  }
846  }
847  }
848  }
849  }
850  }
851  }
852  }
853  }
854  }
855  }
856  }
857  }
858  }
859  }
860  }
861  }
862  }
863  }
864  }
865  }
866  }
867  }
868  }
869  }
870  }
871  }
872  }
873  }
874  }
875  }
876  }
877  }
878  }
879  }
880  }
881  }
882  }
883  }
884  }
885  }
886  }
887  }
888  }
889  }
890  }
891  }
892  }
893  }
894  }
895  }
896  }
897  }
898  }
899  }
900  }
901  }
902  }
903  }
904  }
905  }
906  }
907  }
908  }
909  }
910  }
911  }
912  }
913  }
914  }
915  }
916  }
917  }
918  }
919  }
920  }
921  }
922  }
923  }
924  }
925  }
926  }
927  }
928  }
929  }
930  }
931  }
932  }
933  }
934  }
935  }
936  }
937  }
938  }
939  }
940  }
941  }
942  }
943  }
944  }
945  }
946  }
947  }
948  }
949  }
950  }
951  }
952  }
953  }
954  }
955  }
956  }
957  }
958  }
959  }
960  }
961  }
962  }
963  }
964  }
965  }
966  }
967  }
968  }
969  }
970  }
971  }
972  }
973  }
974  }
975  }
976  }
977  }
978  }
979  }
980  }
981  }
982  }
983  }
984  }
985  }
986  }
987  }
988  }
989  }
990  }
991  }
992  }
993  }
994  }
995  }
996  }
997  }
998  }
999  }
1000  }

```

## 8. OUTPUTS

### ▪ Output1 (Correct Answer):

```
=====
      WELCOME TO KBC QUIZ
=====
Enter Player Name: Tharun Sai
Enter Age: 21

Welcome Tharun Sai (21 years)
-----
Rules:
1. Each question has 5 options
2. Option 5 = Lifeline
3. Lifelines: 50-50 & Audience Poll (Once each)
4. Wrong answer ends the game
-----

-----
1. Which keyword is used to inherit a class in Java?
1) implements
2) inherit
3) extends
4) super
5) Help (Lifeline)
Choose your option: 3
| Correct Answer!
| Congratulations Tharun Sai, your prize is Rs.1000
```

### ▪ Output2 (50-50 Lifeline):

```
2. Which planet has the largest number of moons?
1) Earth
2) Mars
3) Jupiter
4) Saturn
5) Help (Lifeline)
Choose your option: 5
Choose Lifeline:
1) 50-50
2) Audience Poll
1
| 50-50 Lifeline Activated!

2. Which planet has the largest number of moons?
1) Earth
4) Saturn
Choose your option:
```

■ Output3 (Audience poll Lifeline):

```
3. What will be the output of: System.out.println(5 + 2 + "3");
1) 10
2) 73
3) 523
4) 32
5) Help (Lifeline)
Choose your option: 5
Choose Lifeline:
2) Audience Poll
2
Audience Poll:
Option 2 has highest votes!
```

```
-----
3. What will be the output of: System.out.println(5 + 2 + "3");
1) 10
2) 73
3) 523
4) 32
5) Help (Lifeline)
Choose your option:
```

■ Output4 (No lifelines):

```
4. Who directed the movie *Baahubali*?
1) S. S. Rajamouli
2) Shankar
3) Mani Ratnam
4) Atlee
5) Help (Lifeline)
Choose your option: 5
No lifelines left!
```

```
-----
4. Who directed the movie *Baahubali*?
1) S. S. Rajamouli
2) Shankar
3) Mani Ratnam
4) Atlee
5) Help (Lifeline)
Choose your option:
```

▪ Output5 (Wrong Answer) :

4. Who directed the movie \*Baahubali\*?

1) S. S. Rajamouli

2) Shankar

3) Mani Ratnam

4) Atlee

5) Help (Lifeline)

Choose your option: 4

| Wrong Answer!

Game Over Tharun Sai

You won Rs.5000

▪ Output6 (Quiz Won):

-----  
10. What will be the output of: System.out.println("Java".length());

1) 3

2) 4

3) 5

4) 6

5) Help (Lifeline)

Choose your option: 2

Correct Answer!

Congratulations Tharun, your prize is Rs.640000

Amazing Tharun! You completed all questions!

Total Prize: Rs.640000

## 9. LIMITATIONS

- Console-based interface limits visual appeal
- Questions are hardcoded
- No database support
- No multi-user support
- No timer mechanism
- No saving of progress

## 10. CONCLUSION

- The KBCQuiz project successfully demonstrates how core Java concepts can be used to build an engaging and interactive console application. It integrates logical flow, decision-making, and user interaction into a single structured program. The inclusion of lifelines, prize tracking, and colored outputs enhances the realism of the game.
- This project strengthens understanding of Java fundamentals and serves as an excellent beginner-level application. It can be further enhanced by adding a graphical interface, database connectivity, dynamic question loading, and timer-based gameplay.

