1. Create class that accepts file name through constructor, and has method read for processing file line-by-line. **(5 marks)**
2. Create class *SocialMediaPost*, that will be used to store general information about posts on social media. Class should keep information about username, date of post. Create getters and setters for attributes. **(5 marks)**
3. Create class *TwitterPost* that will extend *SocialMediaPost* and will have number of likes and number of retweets in addition to username and date of post. Create getters and setters for new attributes. **(5 marks)**
4. Create class *FacebookPost* that will extend *SocialMediaPost* and will have number of reactions and number of shares in addition to username and date of post. Create getters and setters for new attributes. **(5 marks)**
5. Create class TikTokPost that will extend *SocialMediaPost* and will have number of likes in addition to username and date of post. Create getters and setters for new attributes. **(5 marks)**
6. Create interface *ISocialMediaStatistics* that has one method reactions, that does not accept arguments and return integer. Implement this method in all classes extending from *SocialMediaPost*. **(5 marks)**
   - Twitter formula for reactions is number of likes + number of retweets.
   - *TikTok* number of reactions is equal to number of likes
   - Facebook number of reactions is equal to number of reactions + number of shares * 3,5

In the main program
Use class for reading files from file to process twitter.txt, tiktok.txt and facebook.txt files. And parse lines in those into class instances. Save content of files in a Map, that has key string (social media) and value ArrayList from that social media. **(5 marks)**

1. Calculate average number of reactions per social media and output results into a file. Create a separate fille for each social media, and one for all. **(15 marks)**
2. Find top 10 posts per social media posts based on engagement and output results into a file. **(10 marks)**
3. Find social media posts with most reactions in 2024 and output result into a file. If post has less than 100.000 views don't take it in consideration. **(10 marks)**
4. Create builder design pattern for creating new social media post. **(10 marks)**
5. Find post(s) with most reactions across all social media and save results into a file.
   - **(i)** Solution without threads **(10 marks)**
   - **(ii)** Solution with threads **(20 marks)**

Solution:
Main:

```
import java.io.IOException;
    import java.io.PrintWriter;
    import java.util.*;
    import java.util.stream.Collectors;
```

```
public class Main {
    public static void main(String [] args) {
        FileReaderHelp twitterReader = new FileReaderHelp ("twitter.txt");
        FileReaderHelp facebookReader = new FileReaderHelp ("facebook.txt");
        FileReaderHelp tiktokReader = new FileReaderHelp ("tiktok.txt");

        Map<String, ArrayList<SocialMediaPost>> postsMap = new HashMap<>();
        postsMap.put("Twitter", parseTwitter(twitterReader.read()));
        postsMap.put("Facebook", parseFacebook(facebookReader.read()));
        postsMap.put("TikTok", parseTikTok(tiktokReader.read()));

        writeAverageReactions(postsMap);

        writeTop10Posts(postsMap);

        writeBest2024(postsMap);

    }

    private static ArrayList<SocialMediaPost> parseTwitter(List<String> lines) {
        ArrayList<SocialMediaPost> list = new ArrayList<>();
        for (String line : lines) {
            String[] parts = line.split(",");
            if (parts.length == 4) {
                list.add(new TwitterPost(parts[0], parts[1],
                        Integer.parseInt(parts[2]),
                        Integer.parseInt(parts[3])));
            }
        }
        return list;
    }

    private static ArrayList<SocialMediaPost> parseFacebook(List<String> lines) {
        ArrayList<SocialMediaPost> list = new ArrayList<>();
        for (String line : lines) {
            String[] parts = line.split(",");
            if (parts.length == 4) {
                list.add(new FacebookPost(parts[0], parts[1],
                        Integer.parseInt(parts[2]),
                        Integer.parseInt(parts[3])));
            }
        }
        return list;
    }

    private static ArrayList<SocialMediaPost> parseTikTok(List<String> lines) {
        ArrayList<SocialMediaPost> list = new ArrayList<>();
        for (String line : lines) {
            String[] parts = line.split(",");
            if (parts.length == 3) {
                list.add(new TikTokPost(parts[0], parts[1],
                        Integer.parseInt(parts[2])));
            }
        }
        return list;
    }


    private static void writeAverageReactions(Map<String, ArrayList<SocialMediaPost>>
postsMap) {
        try (PrintWriter allOut = new PrintWriter("average_all.txt")) {
            for (String key : postsMap.keySet()) {
                ArrayList<SocialMediaPost> posts = postsMap.get(key);
                double avg = posts.stream()
                        .mapToInt(p -> ((ISocialMediaStatistics)p).reactions())
                        .average()
                        .orElse(0);
                try (PrintWriter out = new PrintWriter("average_" + key.toLowerCase() + ".txt")) {
                    out.println("Average reactions for " + key + ": " + avg);
                }
                allOut.println("Average reactions for " + key + ": " + avg);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private static void writeTop10Posts(Map<String, ArrayList<SocialMediaPost>>
postsMap) {
        for (String key : postsMap.keySet()) {
            List<SocialMediaPost> top10 = postsMap.get(key).stream()
                    .sorted((a, b) -> Integer.compare(((ISocialMediaStatistics)b).reactions(),
                            ((ISocialMediaStatistics)a).reactions()))
                    .limit(10)
                    .collect(Collectors.toList());
            try (PrintWriter out = new PrintWriter("top10_" + key.toLowerCase() + ".txt")) {
                for (SocialMediaPost p : top10) {
                    out.println(p.toString() + " Reactions=" +
((ISocialMediaStatistics)p).reactions());
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    private static void writeBest2024(Map<String, ArrayList<SocialMediaPost>> postsMap) {
        List<SocialMediaPost> allPosts = postsMap.values().stream()
                .flatMap(Collection::stream)
                .collect(Collectors.toList());

        Optional<SocialMediaPost> best = allPosts.stream()
                .filter(p -> p.getDate().contains("2024"))
                .filter(p -> ((ISocialMediaStatistics)p).reactions() >= 100000)
                .max(Comparator.comparingInt(p -> ((ISocialMediaStatistics)p).reactions()));

        if (best.isPresent()) {
            try (PrintWriter out = new PrintWriter("best2024.txt")) {
                out.println(best.get().toString() + " Reactions=" +
((ISocialMediaStatistics)best.get()).reactions());
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

FileReaderHelp:

```
import java.io.BufferedReader;
    import java.io.FileReader;
    import java.io.IOException;
    import java.util.ArrayList;
    import java.util.List;

    public class FileReaderHelp {
        private String fileName;
        public FileReaderHelp(String fileName) {
            this.fileName = fileName;
        }

        public List<String> read() {
            List<String> lines = new ArrayList<>();
            try {
                BufferedReader br = new BufferedReader(new FileReader(fileName));
                br.readLine();

                String line;
                while ((line = br.readLine()) != null) {
                    lines.add(line);
                }
            }
            catch (IOException e)  {
                System.out.println ("Error reading file: " + e.getMessage());
            }
     return lines;
        }
     }
```

SocialMediaPost:

```
public abstract class SocialMediaPost {
        public String username;
        public String date;

        public SocialMediaPost(String username, String date) {
            this.username = username;
            this.date = date;
        }

        public SocialMediaPost () {}

        public String getUsername() {
            return username;
        }

        public void setUsername (String username) {
            this.username = username;
        }

        public String getDate () {
            return date;
        }

        public void setDate (String date) {
            this.date = date;
        }

     }
```

FacebookPost:

```
public class FacebookPost extends SocialMediaPost  implements ISocialMediaStatistics {
        public int reactions;
        public int reshares;

        public FacebookPost () {
            super("", "");
            this.reactions = 0;
            this.reshares = 0;
        }

        public FacebookPost (String username, String date, int reactions, int reshares) {
            super(username, date);
            this.reactions = reactions;
            this.reshares = reshares;
        }

        @Override
        public int reactions() {
            return (int)(reactions + reshares * 3.5);
        }

        public int getReactions() {
            return reactions;
        }
        public int getShares() {
            return reshares;
        }
        public void setReactions(int reactions) {
            this.reactions = reactions;
        }
        public void setShares(int reshares) {
            this.reshares = reshares;
        }

        @Override
        public String toString() {
            return "FacebookPost{ " +
                    "Username: " + getUsername() +
                    ", date of publishing: " + getDate() +
                    ", number of reactions: " + reactions +
                    ", number of reshares: " + reshares +
                    " }";
        }
    }
```

TikTokPost:

```
public class TikTokPost extends SocialMediaPost  implements ISocialMediaStatistics {
        public int likes;

        public TikTokPost () {
            super("", "");
            this.likes = 0;
        }
```

```
       public TikTokPost (String username, String date, int likes) {
          super( username, date);
          this.likes=likes;
       }

       @Override
       public int reactions() {
          return likes;
       }

       public int getLikes() {
          return likes;
       }

       public void setLikes (int likes) {
          this.likes = likes;
       }

       @Override
       public String toString () {
          return "TikTokPost{ " +
                "Username: " + getUsername() +
                ", date of publishing: " + getDate() +
                ", number of likes: " + likes +
                " }";
       }
    }
```

TwitterPost:

```
public class TwitterPost extends SocialMediaPost  implements ISocialMediaStatistics  {
    public int likes;
    public int retweets;

    public TwitterPost () {
          super("", "");
          this.likes = 0;
          this.retweets = 0;
       }

    public TwitterPost (String username, String date, int likes, int retweets) {
       super(username, date);
       this.likes = likes;
       this.retweets = retweets;
    }

      @Override
    public int reactions() {
        return likes + retweets;
    }

public int getLikes () {
    return likes;
}
public void setLikes (int likes) {
    this.likes=likes;
}

public int getRetweets () {
    return retweets;
}
 public void setRetweets (int retweets) {
    this.retweets=retweets;
 }

 @Override
    public String toString () {
    return "TwitterPost{ " +
            "UserName: " + getUsername() +
            ", date of publishing: " + getDate() +
            ", number of likes: " + likes +
            ", number of retweets: " + retweets +
            " }";
 }
}
```

ISocialMediaStatistics:

```
public abstract interface ISocialMediaStatistics {
        int reactions();
    }
```