```
package program;
import java.util.*;
import java.time.*;

                // ---------- Article ----------
                class Article {
                   private int articleId;
                   private String title;
                   private String category;
                   private String publisher;
                   private LocalDateTime publishTime;

                   public Article(int articleId, String title, String category, String publisher,
LocalDateTime publishTime) {
                      this.articleId = articleId;
                      this.title = title;
                      this.category = category;
                      this.publisher = publisher;
                      this.publishTime = publishTime;
                   }

                   public int getArticleId() { return articleId; }
                   public String getTitle() { return title; }
                   public String getCategory() { return category; }
                   public String getPublisher() { return publisher; }
                   public LocalDateTime getPublishTime() { return publishTime; }

                   @Override
```

```
    public String toString() {
        return "[" + category + "] " + title + " - " + publisher + " (" +
publishTime.toLocalTime() + ")";
    }
}

// ---------- Source ----------
class Source {
    private int sourceId;
    private String name;
    private String category;
    private double trustScore;

    public Source(int sourceId, String name, String category, double
trustScore) {
        this.sourceId = sourceId;
        this.name = name;
        this.category = category;
        this.trustScore = trustScore;
    }

    public int getSourceId() { return sourceId; }
    public String getName() { return name; }
    public String getCategory() { return category; }
    public double getTrustScore() { return trustScore; }
    public void setTrustScore(double trustScore) { this.trustScore =
trustScore; }

    @Override
```

```
    public String toString() {
        return "Source: " + name + " (" + category + ") TrustScore=" +
trustScore;
    }
}

// ---------- Subscriber (Base Class) ----------
abstract class Subscriber {
    private int id;
    private String name;
    private String email;
    private List<String> preferences;
    private String plan;

    public Subscriber(int id, String name, String email, List<String>
preferences, String plan) {
        this.id = id;
        this.name = name;
        this.email = email;
        this.preferences = preferences;
        this.plan = plan;
    }

    public int getId() { return id; }
    public String getName() { return name; }
    public String getEmail() { return email; }
    public List<String> getPreferences() { return preferences; }
    public String getPlan() { return plan; }
```

```
    public abstract List<Article> buildDigest(List<Article> fetchedArticles);
}

// ---------- FreeSubscriber ----------
class FreeSubscriber extends Subscriber {
    public FreeSubscriber(int id, String name, String email, List<String>
preferences) {
        super(id, name, email, preferences, "FREE");
    }

    @Override
    public List<Article> buildDigest(List<Article> fetchedArticles) {
        // Free plan: max 3 articles
        List<Article> digest = new ArrayList<>();
        for (Article a : fetchedArticles) {
            if (getPreferences().contains(a.getCategory())) {
                digest.add(a);
            }
            if (digest.size() == 3) break;
        }
        return digest;
    }
}

// ---------- PaidSubscriber ----------
class PaidSubscriber extends Subscriber {
    public PaidSubscriber(int id, String name, String email, List<String>
preferences) {
        super(id, name, email, preferences, "PAID");
```

```
    }

    @Override
    public List<Article> buildDigest(List<Article> fetchedArticles) {
        // Paid plan: unlimited but filter by preference
        List<Article> digest = new ArrayList<>();
        for (Article a : fetchedArticles) {
            if (getPreferences().contains(a.getCategory())) {
                digest.add(a);
            }
        }
        return digest;
    }
}

// ---------- NewsService ----------
class NewsService {
    private List<Source> sources = new ArrayList<>();
    private List<Article> articles = new ArrayList<>();
    private List<Subscriber> subscribers = new ArrayList<>();

    public void addSource(Source s) { sources.add(s); }
    public void addSubscriber(Subscriber s) { subscribers.add(s); }

    // Simulated fetch
    public void fetchArticles() {
        articles.clear();
        int id = 1;
```

```
            for (Source s : sources) {
                articles.add(new Article(id++, "Latest from " + s.getName(),
                        s.getCategory(), s.getName(),
LocalDateTime.now().minusMinutes(new Random().nextInt(60))));
            }
        }

        // Overloaded filters
        public List<Article> filter(String category) {
            List<Article> res = new ArrayList<>();
            for (Article a : articles) {
                if (a.getCategory().equalsIgnoreCase(category)) res.add(a);
            }
            return res;
        }

        public List<Article> filter(LocalDateTime since) {
            List<Article> res = new ArrayList<>();
            for (Article a : articles) {
                if (a.getPublishTime().isAfter(since)) res.add(a);
            }
            return res;
        }

        public List<Article> filterByKeyword(String keyword) {
            List<Article> res = new ArrayList<>();
            for (Article a : articles) {
                if (a.getTitle().toLowerCase().contains(keyword.toLowerCase()))
res.add(a);
```

```
        }
        return res;
    }

    public void deliverDigests() {
        for (Subscriber s : subscribers) {
            List<Article> digest = s.buildDigest(articles);
            System.out.println("\n--- Digest for " + s.getName() + " (" +
s.getPlan() + ") ---");
            if (digest.isEmpty()) {
                System.out.println("No articles found for preferences.");
            } else {
                for (Article a : digest) {
                    System.out.println(a);
                }
            }
        }
    }

    public void trustReport() {
        System.out.println("\n--- Source Trust Report ---");
        for (Source s : sources) {
            System.out.println(s);
        }
    }
}

// ---------- Main Class ---------
```

```
public class main {

        // TODO Auto-generated method stub

    public static void main(String[] args) {
        NewsService service = new NewsService();

        // Add sources
        service.addSource(new Source(1, "TechCrunch", "Tech", 8.5));
        service.addSource(new Source(2, "ESPN", "Sports", 9.0));
        service.addSource(new Source(3, "Wired", "Tech", 8.0));
        service.addSource(new Source(4, "SkySports", "Sports", 7.5));

        // Add subscribers
        service.addSubscriber(new FreeSubscriber(1, "Alice",
"alice@mail.com", Arrays.asList("Tech")));
        service.addSubscriber(new PaidSubscriber(2, "Bob",
"bob@mail.com", Arrays.asList("Tech", "Sports")));

        // Simulate fetch
        service.fetchArticles();

        // Deliver Digests
        service.deliverDigests();

        // Trust Report
        service.trustReport();

        // Example filtering
```

```
        System.out.println("\n--- Filtered by Category 'Tech' ---");
        for (Article a : service.filter("Tech")) {
            System.out.println(a);
        }
    }

}
```