BÀI TẬP PHẦN 1. CÁC BÀI 6 7 8 ĐƯỢC SỬ DỤNG AI HỖ TRỢ. Viết trong 1 project, mỗi bài trong một package riêng. Project này nếu đẩy lên git phải để ở chế độ private.

BÀI 1. Sử dụng tập hợp (Set) để tìm giao, hợp, hiệu giữa 3 tập hợp số:

Tập A: 5.000.000 số nguyên ngẫu nhiên trong khoảng [1, 10.000.000]
Tập B: 7.000.000 số nguyên ngẫu nhiên trong khoảng [5.000.000, 15.000.000]
Tập C: 6.000.000 số nguyên ngẫu nhiên trong khoảng [8.000.000, 18.000.000]

Tập giao A∩B phải có ít nhất 10% phần tử trùng nhau
Tập giao B∩C phải có ít nhất 15% phần tử trùng nhau
So sánh hiệu suất: HashSet vs TreeSet xem cái nào tốt hơn.
Tính toán (A∪B∪C) - (A∩B∩C) trong thời gian dưới 2 giây

BÀI 2. Cho một văn bản như trong file "input 2.zip". Hãy đọc file và đếm số lần xuất hiện của từng từ (Sử dụng stringtokenizer để tách từ, Các từ này được tách với nhau bằng các dấu sau " .,!=+-") rồi ghi vào file output.txt. Lưu ý, bỏ qua các từ có độ dài < 3 ký tự.

BÀI 3. Lập trình đếm từ đa luồng cho bài 2 với dữ liệu vào là một folder chứa nhiều file text được nén trong file "input 3.zip" bằng cách sử dụng Callable và CompletableFuture. Hãy xử lý song song các file và tìm top 10 từ xuất hiện nhiều nhất, và top 10 từ xuất hiện ít nhất, từ dài nhất, ngắn nhất, trung bình độ dài từ của toàn bộ dữ liệu có trong folder. Lưu ý, chỉ được chạy tối đa 6 luồng cùng lúc. Hiển thị % hoàn thành theo thời gian thực (trên console).

BÀI 4. Tạo ra một class Point với 2 tọa độ nguyên x và y. Sinh ngẫu nhiên 30.000 point khác nhau, nằm trong 3 Set. Set thứ nhất có size 8.000, chứa các điểm có cách điểm A(800, 800) một độ dài không quá 400 đơn vị, Set tiếp theo có size = 10.000 chứa các điểm cách điểm B(4000,800) không quá 500 đơn vị, và 12.000 điểm cuối cùng cách điểm C(2400, 2400) không quá 600 đơn vị. Trộn ngẫu nhiên 30.000 điểm này, sau đó ghi ra file output4.txt. Lưu ý: sử dụng hàm Set< Point >.contains để kiểm tra một điểm đã tồn tại trong tập hợp đó hay chưa (kiểm tra kỹ lại check 2 điểm có tọa độ giống nhau).

BÀI 5. Trong file Maze.java có chứa bản đồ mê cung, hãy implement hàm `solve` dựa trên một phương pháp tìm đường nào đó để tìm đường đi đến điểm đánh dấu ở giữa bản đồ. Hãy chọn các phương pháp có độ phức tạp (số bước thực hiện) càng nhỏ càng tốt và đánh dấu đường đi bằng cách thay số 0 bằng số 2 trên ma trận mê cung. Lưu ý, bài tập này có xét cách viết code, comment, viết java doc, đặt tên biến/hàm, tổ chức project …

BÀI 6. Thiết kế hướng đối tượng
Tạo các đối tượng (class) cho một hệ thống xuất hóa đơn của một cửa hàng tạp hóa. Hệ thống bao gồm các đối tượng sau & các thuộc tính cần xử lý:
KhachHang: mã KH, giới tính, tuổi, loại thành viên (VIP/Regular), điểm tích lũy
NhanVienBanHang: mã NV, giới tính, ngày bắt đầu, ca đăng ký, hoa hồng
NhanVienNhapHang: mã NV, giới tính, ngày bắt đầu, thâm niên, khu vực phụ trách
MatHang: mã hàng, tên, phân loại, giá, tồn kho, nhà cung cấp, hạn sử dụng
HoaDon: mã hóa đơn, NV bán hàng, KH, DS mặt hàng, tổng giá, ngày mua, khuyến mại
Lưu ý: Nhân viên bán hàng và nhân viên nhập hàng có mã nv không trùng nhau và cần được lưu chung vào một danh sách

Tạo ra các file lưu trữ các danh sách sau:
+ danh sách khách hàng (ít nhất 4 khách)
+ nhân viên bán hàng (ít nhất 2) với doanh số khác nhau
+ nv nhập hàng (ít nhất 1)
+ mặt hàng (ít nhất 5)
+ hóa đơn (ít nhất 3)

Tính hoa hồng theo từng bậc cho nhân viên bán hàng:

Bậc 1: 0-10M doanh số → 2% hoa hồng
Bậc 2: 10-25M doanh số → 3.5% hoa hồng
Bậc 3: 25-50M doanh số → 5% hoa hồng
Bậc 4: 50-100M doanh số → 7% hoa hồng
Bậc 5: >100M doanh số → 10% hoa hồng
Tính tích lũy: nhân viên bán 30M sẽ được hoa hồng = (10M×2%) + (15M×3.5%) + (5M×5%)
Hệ số thâm niên: mỗi năm làm việc +10% hoa hồng (tối đa 100%)

Viết method Giảm giá %: VD: "Giảm 15% đồ điện tử, đơn từ 2M, tối đa giảm 500K", các con số kia được đưa vào params.

Viết method tính điểm tĩnh lũy cho khách hàng được tính trên từng hóa đơn. Quy tắc như sau:

- Tích lũy: 1.000đ = 1 điểm (chỉ tính trên giá gốc, không tính trên giá đã giảm)
- Khách VIP: hệ số nhân x1.5 điểm

Yêu cầu:
+ Tạo hóa đơn với áp dụng khuyến mại và tích điểm
+ Tính hoa hồng tháng cho nhân viên

BÀI 7. Một url là url thỏa mãn các yếu tố (ví dụ 1 url hợp lệ: https://tiki.vn/dien-thoai-may-tinh-bang/c1789?src=mega-menu):
- Bắt đầu bằng http hoặc https
- Có thể chứa www hoặc không
- Tên miền của url (ví dụ tiki) chỉ chứa các ký tự la tinh hoa/thường từ a-z và chữ số.
- Phần domain extension (ví dụ .vn) phải chứa dấu chấm ở đầu, thêm sau là các ký tự in thường có độ dài từ 2 đến 6 ký tự.
- Phần path (ví dụ /dien-thoai-may-tinh-bang/c1789?src=mega-menu) không được chứa dấu cách.

Viết đoạn regex để kiểm tra xem một url bất kỳ có là hợp lệ hay không.

BÀI 8. Cho hai lớp Country và City có các thuộc tính (như bên dưới). Một quốc gia có 1 thủ đô và có thể có 1 hoặc nhiều thành phố. Một thành phố luôn trực thuộc chỉ 1 quốc gia.
- Country:
  + String code (mã quốc gia)
  + String name
  + String continent (mã lục địa)
  + double surfaceArea (diện tích bề mặt)
  + int population (dân số)
  + double gnp (Gross National Product)
  + int capital (mã thành phố là thủ đô của đất nước này)
- City:
  + int id
  + String name
  + int population (dân số của thành phố)

- Dữ liệu về các quốc gia, thành phố được lưu trong 2 file countries.dat và cities. Hãy thực hiện đọc thông tin các thành quốc, quốc gia từ file
sau đó sử dụng lambda và stream api trong java 8 để thực hiện các yêu cầu sau:

1.1 Tìm thành phố đông dân nhất của mỗi quốc gia.
1.2 Tìm thành phố đông dân nhất của mỗi lục địa.
1.3 Tìm thành phố là thủ đô, đông dân nhất.
1.4 Tìm thành phố là thủ đô, đông dân nhất của mỗi lục địa.
1.5 Sắp xếp các quốc gia theo số lượng thành phố giảm dần.
1.6 Sắp xếp các quốc gia theo mật độ dân số theo thứ tự giảm dần bỏ qua các quốc gia có dân số bằng không.