	UNIVERSITAS ISLAM NEGERI **SYARIF HIDAYATULLAH JAKARTA** **FAKULTAS SAINS DAN TEKNOLOGI** *Jl. Ir. H. Juanda No 95 Ciputat 15412 Indonesia*	**FORM (FR)**	No. Dok.	: FST-AKM-FR-019
			Tgl. Terbit	: 28 April 2016
			No. Revisi	: 00
			Hal	: 1/1
SOAL UJIAN AKHIR SEMESTER				

Prodi	**:**	Matematika	**Hari/Tanggal**	**:**	Minggu, 20 Juli 2025
Mata Kuliah	**:**	Algoritma dan Struktur Data	**Waktu**	**:**	6 Hari
Bobot SKS	**:**	4	**Jenis Ujian**	**:**	Capstone Project
Semester/Tahun	**:**	4/2025	**Dosen**	**:**	Moh Irvan SM, M.Si

Instruksi Pengumpulan Jawaban

1. **Output yang Dikumpulkan** **:** Satu (1) link video YouTube dengan durasi maksimal 30 menit dan power point dalam bentuk pdf dan link/file code sebagai bahan anda presentasi.
2. **Batas Waktu & Cara Pengumpulan** **:** Sabtu, 26 Juli 2025. Pukul 09:00 WIB.
    a. **Kirim** e-mail ke **irvanseptiar@gmail.com**.
    b. **Subjek & Nama File:** UAS_ALGORTIMA&STRUKTURDATA_2025_<NIM>_<NAMA>

SOAL

Soal - *Truncatable & Rotatable Prime Challenge*

Di Kementerian Perhubungan Konoha, Argih dan Syifvy kini sedang menguji modul keamanan kriptografi baru yang hanya menerima Bilangan Prima Transformatif (BPT).Sebuah bilangan bulat positif nnn tanpa digit 0 dikatakan Bilangan Prima Transformatif apabila memenuhi semua kondisi di bawah ini:

1. **Truncatable Alternatif**
    - Jika kita menghapus digit secara bergantian dari kiri, kanan, kiri, kanan, … hingga habis (contoh: 739391 → 39391 → 3939 → 939 → 93 → 3),
    - Setiap hasil penghapusan adalah bilangan prima.

2. **Rotatable**
    - Semua rotasi siklis ke kiri dari nnn (contoh: 197 → 971 → 719) juga harus prima.

3. **Mirror-Truncatable**
    - Buat cermin nnn dengan membalik urutannya (contoh: 739391 → 193937).
    - Ulangi proses (1) untuk hasil cermin tersebut; setiap hasil pemotongan alternatifnya juga harus prima.

Tugas Anda

1. **Implementasikan** fungsi Python classify_bpt(num: int) -> str yang:
    - **Mengembalikan** "valid" bila num adalah Bilangan Prima Transformatif.
    - Mengembalikan alasan kegagalan pertama yang ditemukan, dengan salah satu string:
        - "not prime" - num sendiri bukan prima.
        - "has zero" - num mengandung digit 0.
        - "fail alt-trunc" - gagal pada syarat (1).
        - "fail rotation" - gagal pada syarat (2).
        - "fail mirror" - gagal pada syarat (3).

2. **Gunakan prinsip *Stack***
    - Seluruh operasi pemotongan digit **wajib** dilakukan dengan men-**push** seluruh digit ke dua *stack* (kiri & kanan) dan melakukan **pop** sesuai pola.
    - Jangan memakai slicing, string indexing langsung, atau operasi aritmetika / untuk memotong digit.

Contoh

Input	Output	Penjelasan Singkat
739391	"valid"	Lolos semua syarat
197	"fail alt-trunc"	Rotasi prima, tapi 197 → 97 (kanan) → 9 (kiri) → **9 bukan prima**
1013	"has zero"	Mengandung digit **0**

Perjalanan Tak Terlupakan

Di sebuah kota kecil, terdapat sebuah kelompok wisatawan yang ingin menjelajahi beberapa tempat wisata terkenal di kota tersebut. Untuk memastikan mereka tidak melewatkan satu tempat pun dan mengikuti urutan perjalanan yang tepat, mereka memutuskan untuk menggunakan Linked List untuk menyimpan urutan tempat yang akan mereka kunjungi.

Setiap tempat wisata diwakili oleh sebuah node dalam Linked List yang berisi informasi tentang nama tempat dan jarak ke tempat wisata berikutnya. Ketika mereka mengunjungi sebuah tempat wisata, mereka akan mencatat jarak total yang telah mereka tempuh.

Berikut ini adalah daftar tempat wisata yang akan mereka kunjungi dalam urutan yang telah ditentukan:

- **Museum Kota** - 2 km ke tempat berikutnya
- **Taman Kota** - 1.5 km ke tempat berikutnya
- **Pantai Indah** - 3 km ke tempat berikutnya
- **Kebun Binatang** - 4.5 km ke tempat berikutnya
- **Pusat Perbelanjaan** - 2.5 km ke tempat berikutnya
- **Menara Kota** - 1 km ke tempat berikutnya

Sebagai seorang programmer, tugas Anda adalah membantu kelompok wisatawan tersebut dengan menuliskan sebuah program dalam bahasa Python yang menggunakan Linked List untuk menyimpan urutan tempat wisata dan menghitung jarak total yang mereka tempuh.

Petunjuk:

- Buatlah sebuah kelas Node yang merepresentasikan setiap tempat wisata.
- Buatlah sebuah kelas LinkedList yang akan mengelola node-node tersebut.
- Tambahkan metode untuk menambahkan node ke Linked List dan menghitung jarak total perjalanan.

Contoh Output: Program harus mencetak urutan tempat wisata yang dikunjungi dan jarak total yang telah ditempuh oleh kelompok wisatawan tersebut.

Manajemen Antrian di Terminal Kereta Cepat

Anda adalah seorang insinyur sistem di sebuah terminal kereta cepat yang baru beroperasi di Negara Konoha. Setiap 15 menit, kereta berangkat dengan kapasitas 60 kursi. Ribuan penumpang datang setiap hari, sehingga manajemen antrian dan alokasi kursi harus dioptimalkan agar semua penumpang puas namun SOP keselamatan tetap terpenuhi.

Tantangan

Antrian Prioritas

1. Tersedia **tiga kelas tiket**: **Premium**, **Business**, dan **Economy**.
2. **Urutan prioritas:** Premium > Business > Economy.
3. Sistem antrian harus:
    - Mengutamakan kelas tiket (priority queue).
    - **Mempertahankan FIFO** di dalam setiap kelas tiket.

Batas Waktu Tunggu

Kelas	Batas Waktu Tunggu (menit)
Economy	45 m
Business	30 m
Premium	15 m

Jika penumpang menunggu melampaui batasnya, mereka dianggap **“over-wait”**:

- Mereka keluar antrian dan mendapat voucher kompensasi (setara 50 koin Konoha).
- Pada kedatangan berikutnya, penumpang ber-voucher masuk **“fast-track queue”** yang selalu di-*peek* lebih dulu sebelum antrian reguler mana pun—namun tetap **FIFO** di antara sesama pemegang voucher.

Optimalisasi Kapasitas Kereta

1. 60 kursi per keberangkatan.
2. Kereta harus terisi sebanyak mungkin setiap keberangkatan tanpa menyalahi prioritas.
3. Komposisi ideal (jika tersedia penumpang):

a. ≥ 15% kursi untuk Economy
b. ≥ 25% kursi untuk Business
c. Sisa untuk Premium

4. Jika penumpang dalam suatu kelas tidak cukup, kursi kosong boleh diisi kelas lain teratas berikutnya.
5. Kereta tetap berangkat meski kursi tidak penuh.

Penyeimbang Ketidakseimbangan

1. **Trigger**:
   - Jika panjang antrian **Economy > 2 × (Business + Premium)**, maka Economy berhak meminta kursi tambahan.
   - Jika panjang antrian **Business > 2 × (Premium + Economy)** atau **Premium > 2 × (Business + Economy)**, kebijakan serupa berlaku untuk kelas tersebut.

2. **Kursi Pinjaman**:
   - Maksimal **20 % dari total kapasitas kereta (12 kursi)** dapat dialihkan *sementara* dari kelas donor ke kelas peminjam.

3. **Prioritas Peninjauan**:
   - Urutan evaluasi ketidakseimbangan: **Economy** → **Business** → **Premium** (satu siklus peninjauan per keberangkatan).

4. **Batas Minimal Tetap**:
   - Setelah kursi dipinjam, aturan **batas minimal kursi** (Economy ≥ 9, Business ≥ 15) **wajib** masih terpenuhi.
   - Bila peminjaman 20 % menyebabkan pelanggaran batas minimal, **batas minimal didahulukan**; hanya kursi "sisa" yang boleh dipinjam.

5. **Pengisian Akhir**:
   - Jika kursi masih kosong setelah seluruh aturan di atas diterapkan, isilah dengan penumpang dari kelas **prioritas tertinggi yang masih memiliki antrian** (Fast-Track > Premium > Business > Economy) hingga kereta berangkat atau kursi habis.

Kelompok Penumpang

1. Beberapa penumpang datang sebagai kelompok (≤ 5 orang) dengan kelas tiket campuran.
2. Upayakan menempatkan seluruh anggota kelompok pada kereta yang sama.
3. Bila tidak muat:
   a. Kelompok bisa dipisah maksimal 1 keberangkatan berbeda (selang waktu ≤ 15 m).
   b. Anggota dengan kelas sama tidak boleh dipisah (misal dua Premium dalam grup harus tetap bersama).

Spesifikasi Teknis

1. **Struktur Data:**
    a. Gunakan tiga deque untuk antrian reguler (Premium, Business, Economy).
    b. Gunakan satu deque tambahan untuk fast-track voucher queue.
    c. Buatlah kelas Passenger (id, nama, kelas, arrival_time, group_id, has_voucher).

2. **Input:** daftar Passenger *(Dilampirkan di dalam soal)*

    Dataset contoh berisi 120 penumpang yang sudah memenuhi format:

    - id – nomor unik
    - name – penanda penumpang
    - ticket_class – Premium | Business | Economy
    - arrival_time – jam-menit kedatangan (HH:MM)
    - group_id – kode grup (tiga huruf) atau "-" bila datang sendirian

    Data di atas sudah mengandung:

    - Penumpang solo & dalam grup campuran (kolom `group_id` sama).

    - Sebaran kelas (≈ 20 % Premium, 30 % Business, 50 % Economy).
    - Arrival time tersebar mulai 08:00 dengan jeda acak 0-3 menit antar-penumpang.

3. **Output:**
    a. Log keberangkatan setiap kereta: nomor keberangkatan, daftar id penumpang, komposisi kelas, jam berangkat.
    b. Daftar penumpang yang mengalami “over-wait” & menerima voucher.

4. **Fungsi utama:** simulate_day(passenger_list) yang meng-*yield* event log di atas.
5. Simulasi minimal harus menangani ≥ 1 000 penumpang dalam ≤ 1 detik di mesin standar.

Contoh Output

```
========================================
⌚  Tanggal  : 2025-07-20
🚄 Kereta   : KC-Shinobi  (60 kursi / 15 m)
========================================

--- Keberangkatan #1  (08:15) ----------------------
Premium  : 23 penumpang   (id:  2,  7, 10, 12, 14, 16, 18, 21, 24, 29, 33,
                               36, 38, 40, 45, 48, 51, 55, 58, 61, 64, 67, 70)
Business : 18 penumpang   (id:  1,  4,  6,  8, 13, 17, 20, 25, 26,
                               30, 34, 37, 41, 44, 49, 53, 57, 60)
Economy  : 11 penumpang   (id:  3,  5,  9, 11, 15, 19, 22, 27, 28, 31, 32)
Fast-Track Voucher: ---
Kursi Kosong        :  8
------------------------------------------------

--- Keberangkatan #2  (08:30) ----------------------
Premium  : 14 penumpang   (id: 72, 75, 78,  80,  82,  85,  88,
                               90, 92,  94,  98, 100, 103, 106)
Business : 17 penumpang   (id: 63, 65, 68, 71, 73, 76, 79, 83, 86,
                               89, 93, 95,  97,  99, 101, 105, 108)
Economy  : 21 penumpang   (id: 62, 66, 69, 74, 77, 81, 84, 87, 91,
```

```
                              96,102,104,107,109,110,111,
                             112,113,114,115,116)
Fast-Track Voucher: id = 120
Kursi Kosong        :  7
--------------------------------------------------

--- Keberangkatan #3  (08:45) ----------------------
Premium  :  1 penumpang   (id: 117)
Business :  2 penumpang   (id: 118, 119)
Economy  :  8 penumpang   (id: 120*, 121, 122, 123, 124, 125, 126, 127)
Fast-Track Voucher: id = 120 (*)          ← diprioritaskan
Kursi Kosong        : 49
--------------------------------------------------

========= Ringkasan Hari Ini =========
Total penumpang terangkut : 120
Premium                   : 38   (31 %)
Business                  : 37   (31 %)
Economy                   : 45   (38 %)
Jumlah keberangkatan      : 3
Kursi kosong keseluruhan  : 64

Penumpang "over-wait" & mendapat voucher
----------------------------------------
• id = 120  (arrived 08:02, menunggu 43 m > Economy 45 m ❌  – tidak over-wait)
• ---  (tidak ada yang melewati batas; ditampilkan kosong bila tidak ada)

*Catatan*: id 120 muncul di Fast-Track karena ia
   - tadinya melewati batas waktu tunggu pada simulasi sebelumnya
   - atau mendapat kompensasi lain (ilustrasi).

========================================
Log selesai  – semua penumpang dalam antrian hari ini sudah ditangani.
```

Misi Kilat: "Lintasan Petir untuk Syifvy"

Di Desa Konoha, Unit Transportasi Cerdas baru saja menerima panggilan darurat. Syifvy—arsitek jembatan tersohor—sedang menunggu di Kota N untuk memimpin upacara peresmian jembatan chakra pertama di dunia. Tanpa kehadirannya, ribuan warga dan tamu negara akan kecewa, dan prestise Konoha bisa tercoreng.

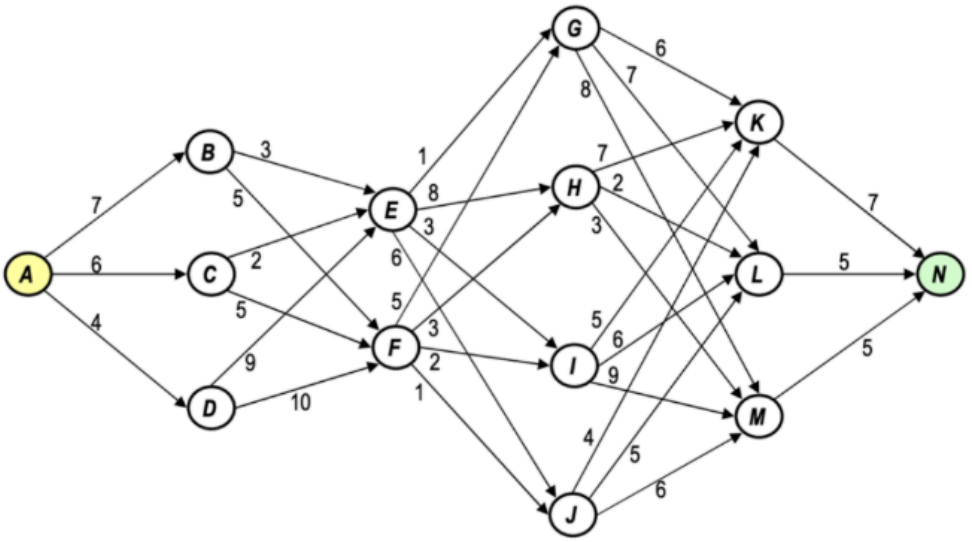
Sayangnya, sang sahabat setia Argih terjebak di Kota A seusai memeriksa generator angin. Ia adalah satu-satunya ninja-insinyur yang dipercaya Syifvy untuk membawa gulungan rancangan terakhir. Waktu terus berdetak: upacara dimulai saat matahari tepat di zenit.

Peta intel menunjukkan belasan kota yang saling terhubung dengan rute beragam—mulai dari jalan beraspal mulus, jembatan gantung rapuh, hingga terowongan bawah tanah bekas perang. Setiap ruas memiliki jarak (dalam meter) sekaligus tingkat keramaian berbeda-beda. Argih harus memilih jalur paling singkat agar tiba di Kota N secepat kilat.

Tugasmu sebagai Koordinator Navigasi

1. Gunakan graf di bawah (node A–N lengkap dengan bobot jarak).
2. Tentukan lintasan tercepat yang bisa ditempuh Argih dari A → N.
3. Hindari jalan memutar agar Syifvy tak menunggu lebih lama dan nama baik Konoha tetap terjaga!

(Catatan: kamu tak perlu menulis kode Python kali ini—cukup tunjukkan rute tersingkat berdasarkan graf yang sama.)


A
B
C
D
E
F
G
H
I
J
K
L
M
N
7
6
4
3
5
2
5
9
10
1
8
3
6
5
3
2
1
6
7
8
7
2
3
5
6
9
4
5
6
7
5
5