



PROGRAM STUDI S1 MATEMATIKA

LAPORAN TUGAS

Struktur Data dan Algoritma

Oleh:

- | | |
|----------------------------|-------------|
| 1. Citra Widya Ningsih | 23030214007 |
| 2. Devita Natalia | 23030214022 |
| 3. Nur Nailil Farichah | 23030214032 |
| 4. Intan Widya Nur 'Azizah | 23030214075 |
| 5. Lailatu Zahro Nataly | 23030214152 |
| 6. Muhammad Arifian | 23030214156 |

MA - 2023E

Dosen Pengampu:

Riska Wahyu ROMADHONIA, S.Si., M.Sc.

Ko-Asisten:

Muhammad Mahdy Al Akbar

12 Juni 2025

1 Permasalahan

Di era digital yang berkembang pesat, kebutuhan akan sistem informasi yang efektif, efisien, dan mudah diakses telah menjadi bagian penting dalam berbagai sektor, termasuk dalam industri hiburan seperti bioskop. Pengelolaan jadwal penayangan film menjadi salah satu aspek krusial yang menentukan kelancaran operasional dan kualitas pelayanan kepada penonton.

Dalam praktiknya, bioskop sering kali menghadapi tantangan dalam menyusun jadwal tayang film yang dinamis, karena terdapat berbagai faktor yang perlu dipertimbangkan seperti durasi film, ketersediaan studio, permintaan penonton, dan jadwal rilis film terbaru. Oleh karena itu, diperlukan sebuah sistem digital yang mampu membantu admin bioskop dalam menyusun, memperbarui, dan mengelola jadwal tayang dengan lebih sistematis dan real-time.

Program manajemen jadwal bioskop yang kami kembangkan hadir sebagai solusi dari permasalahan tersebut. Program ini dirancang dengan fokus utama pada peran admin bioskop sebagai pihak yang bertanggung jawab dalam mengatur data penayangan film. Fitur-fitur utama seperti penambahan dan pengeditan jadwal film, penghapusan film yang sudah tidak tayang, penyortiran berdasarkan jam tayang dan judul film, kemudian pencarian film memanfaatkan struktur data dinamis seperti Linked List dan Queue yang memungkinkan pengolahan data yang efisien, fleksibel, dan sesuai dengan kebutuhan nyata di lapangan.

Dengan sistem ini, admin tidak hanya dapat menyusun jadwal film secara otomatis berdasarkan waktu tayang, tetapi juga dapat memantau dan memperbarui jadwal setiap harinya dengan kemudahan akses. Hal ini tentu membantu mengurangi risiko kesalahan penjadwalan, meningkatkan produktivitas kerja, serta memberikan pengalaman terbaik bagi para penonton yang mengandalkan ketepatan informasi jadwal dalam merencanakan kunjungan mereka ke bioskop.

2 Source Code

Berikut adalah *source code* dari program:

```

1 import json
2 from datetime import datetime, time, timezone, timedelta, date
3
4 class JadwalFilm:
5     def __init__(self, judul, jam: time):
6         self.judul = judul
7         self.jam = jam
8         self.next = None
9
10 class LinkedList:
11     def __init__(self):
12         self.head = None
13
14     def insert(self, judul, jam: time):
15         new_node = JadwalFilm(judul, jam)
16         if self.head is None:
17             self.head = new_node
18         else:
19             current = self.head
20             while current.next:
21                 current = current.next
22             current.next = new_node
23
24     def to_array(self):
25         data = []
26         current = self.head
27         while current:
28             data.append((current.judul, current.jam))
29             current = current.next
30         return data
31
32     def from_array(self, data):
33         self.head = None
34         for item in data:
35             self.insert(item[0], item[1])
36
37     def sort_by(self, key='judul'):
38         data = self.to_array()
39         # Insertion Sort
40         for i in range(1, len(data)):
41             key_item = data[i]
42             j = i - 1
43             while j >= 0 and (
44                 (key == 'judul' and data[j][0].lower() > key_item
45                  [0].lower()) or
46                 (key != 'judul' and data[j][1] > key_item[1])
47             ):
48                 data[j + 1] = data[j]
49                 j -= 1

```

```

49         data[j + 1] = key_item
50     self.from_array(data)
51
52     def hapus_by_index_sorted_judul(self, index):
53         data = self.to_array()
54         data.sort(key=lambda x: x[0].lower())
55         while not (0 < index <= len(data)):
56             print("                Tolong masukkan nomor yang valid.")
57             print("-----")
58             index = int(input("Masukkan nomor film yang ingin
                    dihapus: "))
59         else:
60             target = data[index - 1]
61             tanggal_input = date.today().isoformat()
62             new_data = [item for item in self.to_array() if not (
63                 item[0] == target[0] and item[1] == target[1])]
64             self.from_array(new_data)
65             simpan_ke_history("history_jadwal_bioskop.json", [(
66                 target[0], target[1], tanggal_input)])
67             print(f"Film nomor {index} berhasil dihapus.")
68             return new_data
69
70     def tampilkan_dengan_nomor(self, by='judul'):
71         current = self.head
72         if not current:
73             print("Daftar kosong.")
74             return
75         i = 1
76         while current:
77             if by == 'judul':
78                 print(f" {i}. {current.judul} - {current.jam.
79                     strftime('%H:%M')} WIB")
80             else:
81                 print(f" {i}. {current.jam.strftime('%H:%M')} WIB -
82                     {current.judul}")
83             current = current.next
84             i += 1
85
86     class Queue:
87         def __init__(self):
88             self.items = []
89
90         def enqueue(self, judul, jam: time):
91             tanggal_input = date.today().isoformat()
92             self.items.append((judul, jam, tanggal_input))
93
94         def dequeue_expired(self, current_time: time):
95             new_expired = []
96             remaining = []
97             try:

```

```

94         with open("history_jadwal_bioskop.json", "r") as f:
95             history = json.load(f)
96     except FileNotFoundError:
97         history = {}
98
99     for judul, jam, tgl in self.items:
100         jam_str = jam.strftime('%H:%M')
101         # Jangan masukkan jika sudah ada dalam history tanggal
102         # itu
103         if jam <= current_time:
104             if tgl not in history or (judul, jam_str) not in
105                 history[tgl]:
106                 new_expired.append((judul, jam, tgl))
107         else:
108             remaining.append((judul, jam, tgl))
109
110     self.items = remaining
111     if new_expired:
112         simpan_ke_history("history_jadwal_bioskop.json",
113             new_expired)
114     return new_expired
115
116 def from_array(self, data):
117     self.items = []
118     for judul, jam in data:
119         self.enqueue(judul, jam)
120
121 # === File JSON ===
122 def simpan_ke_json(nama_file, data):
123     data_json = [(judul, jam.strftime('%H:%M')) for judul, jam in
124         data]
125     with open(nama_file, 'w') as f:
126         json.dump(data_json, f)
127
128 def baca_dari_json(nama_file):
129     try:
130         with open(nama_file, 'r') as f:
131             data = json.load(f)
132             return [(judul, datetime.strptime(jam, '%H:%M').time())
133                 for judul, jam in data]
134     except FileNotFoundError:
135         return []
136
137 def simpan_ke_history(nama_file, data_baru):
138     try:
139         with open(nama_file, 'r') as f:
140             data_lama = json.load(f)
141     except FileNotFoundError:
142         data_lama = {}

```

```

139     for j, t, tanggal_input in data_baru:
140         if tanggal_input not in data_lama:
141             data_lama[tanggal_input] = []
142
143             jam_str = t.strftime('%H:%M')
144             if (j, jam_str) not in data_lama[tanggal_input]:
145                 data_lama[tanggal_input].append((j, jam_str))
146
147     with open(nama_file, 'w') as f:
148         json.dump(data_lama, f)
149
150 def update_history():
151     waktu_sekarang = datetime.now(timezone(timedelta(hours=7))).
152         time()
153     expired_data = antrian.dequeue_expired(waktu_sekarang)
154     # Buat linked list hanya dari data yang masih valid
155     linkedlist.from_array([(j, t) for j, t, _ in antrian.items])
156     simpan_ke_json("jadwal_bioskop.json", linkedlist.to_array())
157
158 def baca_history_dengan_tanggal(nama_file):
159     try:
160         with open(nama_file, 'r') as f:
161             return json.load(f)
162     except FileNotFoundError:
163         return {}
164
165 def validasi_waktu(jam):
166     try:
167         h, m = map(int, jam.split(':'))
168         if 0 <= h < 24 and 0 <= m < 60:
169             return True
170         return False
171     except:
172         return False
173
174 # === Main Program ===
175 linkedlist = LinkedList()
176 antrian = Queue()
177
178 data_awal = baca_dari_json("jadwal_bioskop.json")
179 for judul, jam in data_awal:
180     linkedlist.insert(judul, jam)
181
182 linkedlist.sort_by('jam')
183 data_sorted = linkedlist.to_array()
184 linkedlist.from_array(data_sorted)
185 antrian.from_array(data_sorted)
186
187 waktu_sekarang = datetime.now(timezone(timedelta(hours=7))).time()

```

```

188 antrian.dequeue_expired(waktu_sekarang)
189 linkedlist.from_array([(j, t) for j, t, _ in antrian.items])
190 simpan_ke_json("jadwal_bioskop.json", linkedlist.to_array())
191
192 def menu():
193     print("\n\n=====")
194     waktu_sekarang = datetime.now(timezone(timedelta(hours=7))).
195         time()
196     print(f"||      Waktu Saat Ini: {waktu_sekarang.strftime('%H:%M
197         ')} WIB      ||")
198     print("=====")
199     print("||      MENU   BIOSKOP      ||")
200     print("=====")
201     print("1. Daftar Film      ")
202     print("2. Cari Jadwal Film")
203     print("3. Tambah Jadwal Film")
204     print("4. Edit Film      ")
205     print("5. Hapus Jadwal Film")
206     print("6. Lihat History Film Lewat")
207     print("0. Keluar")
208     print("-----")
209
210 while True:
211     menu()
212     pilih = input("Pilih menu: ")
213     update_history()
214
215     if pilih == '1':
216         print("\n=====")
217         print("||      Lihat Jadwal Film      ||")
218         print("=====")
219         print("1. Lihat daftar berdasarkan Judul")
220         print("2. Lihat daftar berdasarkan Jam Tayang")
221
222         opsi = input("Pilih opsi (1/2): ")
223         while (opsi != '1' and opsi != '2'):
224             # kalau opsi bukan 1/2, masukkan opsi lain
225             print("Pilihan tidak valid.")
226             print("-----")
227             print("1. Lihat daftar berdasarkan Judul")
228             print("2. Lihat daftar berdasarkan Jam Tayang")
229             opsi = input("Pilih opsi (1/2): ")
230
231         if opsi == '1':
232             print("=====")
233             print("||  Daftar Film (Berdasarkan Judul)  ||")
234             print("=====")
235             linkedlist.sort_by('judul')
236             linkedlist.tampilkan_dengan_nomor('judul')
237         elif opsi == '2':

```

```

236         print("=====")
237         print("||  Daftar Film (Berdasarkan Jam Tayang)  ||")
238         print("=====")
239         linkedlist.sort_by('jam')
240         linkedlist.tampilkan_dengan_nomor('jam')
241         print("=====")
242
243     elif pilih == '2':
244         print("\n=====")
245         print("||          Cari Jadwal Film          ||")
246         print("=====")
247         print("1. Cari berdasarkan Judul")
248         print("2. Cari berdasarkan Jam Tayang")
249         opsi = input("Pilih jenis pencarian (1/2): ")
250         while (opsi != '1' and opsi != '2'):
251             # kalau opsi bukan 1/2, masukkan opsi lain
252             print("          Pilihan tidak valid.")
253             print("-----")
254             print("1. Cari berdasarkan Judul")
255             print("2. Cari berdasarkan Jam Tayang")
256             opsi = input("Pilih jenis pencarian (1/2): ")
257         found = False
258         data = linkedlist.to_array()
259
260         if opsi == '1': #judul
261             keyword = input("          Masukkan kata kunci judul film: ")
262             )
263             for judul, jam in data: # untuk mengecek aja
264                 if keyword.lower() in judul.lower():
265                     found = True
266             if found: # kalau ditemukan, beri pesan ini
267                 print("-----")
268                 print(f"          Ditemukan film dengan kata kunci {
269                     keyword}")
270             i=1
271             for judul, jam in data: # munculkan data filmnya,
272                 dengan nomer
273                 if keyword.lower() in judul.lower():
274                     print(f" {i}. {judul} - {jam.strftime('%H:%M')}
275                         WIB")
276                     i = i+1
277                     found = True
278             elif opsi == '2': #jam
279                 keyword = input("          Masukkan kata kunci jam tayang (
280                     HH:MM): ")
281                 try:
282                     jam_dicari = datetime.strptime(keyword, '%H:%M').
283                         time()
284                     for judul, jam in data: # mengecek aja
285                         if jam_dicari == jam:

```



```

280         found = True
281     if found:
282         print("-----")
283         print(f"        Ditemukan film dengan jam tayang {
                keyword} WIB")
284     i=1
285     for judul, jam in data:
286         if jam_dicari == jam:
287             print(f" {i}. {judul} - {jam.strftime('%H:%
                M')} WIB")
288             found = True
289     except:
290         print("Format jam tidak valid.")
291 if not found:
292     print("-----")
293     print("        Film tidak ditemukan.")
294     print("=====) # penutup
295
296 elif pilih == '3':
297     print("\n=====")
298     print("||        Tambahkan Jadwal Film        ||")
299     print("=====")
300     judul = input("Judul Film: ")
301     jam_input = input("Jam Tayang (HH:MM): ")
302     while not validasi_waktu(jam_input): # mengecek kevalidan
303         waktu
304         print("        Tolong masukkan waktu yang valid.")
305         print("-----")
306         jam_input = input("Jam Tayang (HH:MM): ")
307
308     jam = datetime.strptime(jam_input, '%H:%M').time()
309     waktu_sekarang = datetime.now(timezone(timedelta(hours=7)))
310     .time()
311     if jam <= waktu_sekarang:
312         print("        Jadwal tidak bisa ditambahkan karena waktu
                sudah terlewat.")
313         print("=====")
314         continue # langsung keluar
315
316     linkedlist.insert(judul, jam)
317     linkedlist.sort_by('jam')
318     data = linkedlist.to_array()
319     antrian.from_array(data)
320     simpan_ke_json("jadwal_bioskop.json", data)
321     print("        Jadwal berhasil ditambahkan!")
322     print("=====")
323
324 elif pilih == '4':
325     linkedlist.sort_by('judul')

```

```

324 print("\n=====")
325 print("||          Daftar Film - Edit          ||")
326 print("=====")
327 data = linkedlist.to_array()
328 for idx, (judul, jam) in enumerate(data, 1):
329     print(f" {idx}. {judul} - {jam.strftime('%H:%M')} WIB")
330 try:
331     print("-----")
332     index = int(input("Masukkan nomor film yang ingin
333                        diedit: "))
334     while not (1 <= index <= len(data)):
335         print("          Tolong masukkan nomor yang valid.")
336         print("-----")
337         index = int(input("Masukkan nomor film yang ingin
338                           diedit: "))
339
340     judul_baru = input("Judul Film Baru: ")
341     jam_baru_input = input("Jam Tayang Baru (HH:MM): ")
342     while not validasi_waktu(jam_baru_input): # mengecek
343         kevalidan waktu
344         print("          Tolong masukkan waktu yang valid.")
345         print("-----")
346         jam_baru_input = input("Jam Tayang Baru (HH:MM): ")
347
348     jam_baru = datetime.strptime(jam_baru_input, '%H:%M').
349         time()
350     waktu_sekarang = datetime.now(timezone(timedelta(hours
351         =7))).time()
352     if jam_baru <= waktu_sekarang: # jika input-an waktu
353         baru, sudah terlewat
354         print("Tidak dapat mengganti dengan jam yang sudah
355             terlewat.")
356         continue
357
358     data[index - 1] = (judul_baru, jam_baru)
359     linkedlist.from_array(data)
360     linkedlist.sort_by('jam')
361     antrian.from_array(linkedlist.to_array())
362     simpan_ke_json("jadwal_bioskop.json", linkedlist.
363         to_array())
364     print("Jadwal berhasil diedit!")
365 except ValueError:
366     print("Input tidak valid.")
367 print("=====")
368
369 elif pilih == '5':
370     linkedlist.sort_by('judul')
371     print("\n=====")
372     print("||          Daftar Film yang Tersedia          ||")
373     print("=====")

```

```

366 data = linkedlist.to_array()
367 for idx, (judul, jam) in enumerate(data, 1):
368     print(f" {idx}. {judul} - {jam.strftime('%H:%M')} WIB")
369 try:
370     print("-----")
371     index = int(input("Masukkan nomor film yang ingin
372         dihapus: "))
373     new_data = linkedlist.hapus_by_index_sorted_judul(index
374         )
375     linkedlist.from_array(new_data)
376     linkedlist.sort_by('jam')
377     antrian.from_array(new_data)
378     simpan_ke_json("jadwal_bioskop.json", new_data)
379 except ValueError:
380     print("Input tidak valid.")
381 print("=====")
382
383 elif pilih == '6':
384     print("\n=====")
385     print("||          Riwayat Film          ||")
386     print("=====")
387     history = baca_history_dengan_tanggal("
388         history_jadwal_bioskop.json")
389     if not history:
390         print("Belum ada history.")
391     else:
392         for tgl, daftar in sorted(history.items(), reverse=True
393             ):
394             print(f"Tanggal: {tgl}")
395             print("Riwayat Film:")
396             for idx, (judul, jam) in enumerate(daftar, 1):
397                 print(f" {idx}. {judul} - {jam} WIB")
398             print("=====")
399
400 elif pilih == '0':
401     print("\n=====")
402     print("||  Terima kasih telah menggunakan program ini.  ||")
403     print("=====\\n
404         ")
405     break
406
407 else:
408     print("          PILIHAN TIDAK VALID")
409     print("=====")
410
411 print("- Admin      ")

```

3 Tangkapan Layar Hasil *Running Source Code*

3.1 *Tampilan awal*

```
=====
||      Waktu Saat Ini: 21:44 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📅
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu:
```

Gambar 1: Tampilan awal

3.2 *Pengguna memilih opsi 1 (Melihat Daftar Film)*

```
=====
||      Waktu Saat Ini: 23:54 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📅
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 1

=====
||      Lihat Jadwal Film      ||
=====
1. Lihat daftar berdasarkan Judul
2. Lihat daftar berdasarkan Jam Tayang
Pilih opsi (1/2): 1
=====
||  Daftar Film (Berdasarkan Judul)  ||
=====
1. Keluarga Cemara - 23:55 WIB
2. Pandora - 23:56 WIB
=====
- Admin ❤️
```

Gambar 2: Daftar film berdasarkan judul

```

=====
||      Waktu Saat Ini: 22:15 WIB      ||
=====
||      MENU   BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👉
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 1

=====
||      Lihat Jadwal Film      ||
=====
1. Lihat daftar berdasarkan Judul
2. Lihat daftar berdasarkan Jam Tayang
Pilih opsi (1/2): 2
=====
||  Daftar Film (Berdasarkan Jam Tayang)  ||
=====
1. 22:30 WIB - Pandora
2. 22:45 WIB - Jumbo
3. 23:00 WIB - Keluarga Cemara
4. 23:00 WIB - Labubu
=====
- Admin ❤️

```

Gambar 3: Daftar film berdasarkan jam tayang

```

=====
||      Waktu Saat Ini: 21:44 WIB      ||
=====
||      MENU   BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👉
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 1

=====
||      Lihat Jadwal Film      ||
=====
1. Lihat daftar berdasarkan Judul
2. Lihat daftar berdasarkan Jam Tayang
Pilih opsi (1/2): 1
=====
||  Daftar Film (Berdasarkan Judul)  ||
=====
Daftar kosong.
=====
- Admin ❤️

```

Gambar 4: Daftar film berdasarkan judul dengan daftar film kosong

```

=====
||      Waktu Saat Ini: 21:44 WIB      ||
=====
||      MENU   BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👍
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 1

=====
||      Lihat Jadwal Film      ||
=====
1. Lihat daftar berdasarkan Judul
2. Lihat daftar berdasarkan Jam Tayang
Pilih opsi (1/2): 2
-----
||  Daftar Film (Berdasarkan Jam Tayang)  ||
=====
Daftar kosong.
=====
- Admin ❤️

```

Gambar 5: Daftar film berdasarkan jam tayang dengan daftar film kosong

3.3 Pengguna memilih opsi 2 (Mencari Film)

```

=====
||      Waktu Saat Ini: 21:49 WIB      ||
=====
||      MENU   BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👍
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 2

=====
||      Cari Jadwal Film      ||
=====
1. Cari berdasarkan Judul
2. Cari berdasarkan Jam Tayang
Pilih jenis pencarian (1/2): 1
👍 Masukkan kata kunci judul film: j
-----
👍 Ditemukan film dengan kata kunci j
1. Jumbo - 22:45 WIB
=====
- Admin ❤️

```

Gambar 6: Mencari film berdasarkan judul

```

=====
||      Waktu Saat Ini: 21:45 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👉
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 2

=====
||      Cari Jadwal Film      ||
=====
1. Cari berdasarkan Judul
2. Cari berdasarkan Jam Tayang
Pilih jenis pencarian (1/2): 1
👉 Masukkan kata kunci judul film: q
-----
😞 Film tidak ditemukan.
=====
- Admin ❤️

```

Gambar 7: Mencari film berdasarkan judul tetapi film tidak ditemukan

```

=====
||      Waktu Saat Ini: 22:02 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👉
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 2

=====
||      Cari Jadwal Film      ||
=====
1. Cari berdasarkan Judul
2. Cari berdasarkan Jam Tayang
Pilih jenis pencarian (1/2): 2
👉 Masukkan kata kunci jam tayang (HH:MM): 22:30
-----
👉 Ditemukan film dengan jam tayang 22:30 WIB
1. Pandora - 22:30 WIB
=====
- Admin ❤️

```

Gambar 8: Mencari film berdasarkan jam tayang

```

=====
||      Waktu Saat Ini: 22:07 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🖋️
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 2

=====
||      Cari Jadwal Film      ||
=====
1. Cari berdasarkan Judul
2. Cari berdasarkan Jam Tayang
Pilih jenis pencarian (1/2): 2
🖋️ Masukkan kata kunci jam tayang (HH:MM): 10.00
Format jam tidak valid.

-----
😞 Film tidak ditemukan.
=====
- Admin ❤️

```

Gambar 9: Mencari film berdasarkan jam tayang, tetapi film tidak ditemukan karena format jam salah

```

=====
||      Waktu Saat Ini: 22:02 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🖋️
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 2

=====
||      Cari Jadwal Film      ||
=====
1. Cari berdasarkan Judul
2. Cari berdasarkan Jam Tayang
Pilih jenis pencarian (1/2): 2
🖋️ Masukkan kata kunci jam tayang (HH:MM): 22:10
-----
😞 Film tidak ditemukan.
=====
- Admin ❤️

```

Gambar 10: Mencari film berdasarkan jam tayang, tetapi film tidak ditemukan

3.4 Pengguna memilih opsi 3 (Menambah jadwal film)

```
=====
||      Waktu Saat Ini: 22:08 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 3

=====
||      Tambahkan Jadwal Film      ||
=====
Judul Film: Labubu
Jam Tayang (HH:MM): 23:00
😄 Jadwal berhasil ditambahkan!
=====
- Admin ❤️
```

Gambar 11: jadwal film berhasil ditambahkan

```
=====
||      Waktu Saat Ini: 22:11 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 3

=====
||      Tambahkan Jadwal Film      ||
=====
Judul Film: Rumah Kaca
Jam Tayang (HH:MM): 21:00
😞 Jadwal tidak bisa ditambahkan karena waktu sudah terlewat.
=====
```

Gambar 12: jadwal film tidak bisa ditambahkan karena waktu sudah terlewat

3.5 Pengguna memilih opsi 4 (Edit jadwal film)

```
=====
||      Waktu Saat Ini: 00:02 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 4

=====
||      Daftar Film - Edit      ||
=====
1. LLS - 00:08 WIB
2. MIB - 00:15 WIB
3. MPH - 00:10 WIB
-----
Masukkan nomor film yang ingin diedit: 3
Judul Film Baru: MDR
Jam Tayang Baru (HH:MM): 00:10
Jadwal berhasil diedit!
=====
- Admin ❤️
```

Gambar 13: Mengedit jadwal film berdasarkan judul

```
=====
||      Waktu Saat Ini: 00:01 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 4

=====
||      Daftar Film - Edit      ||
=====
1. LLS - 00:05 WIB
2. MIB - 00:15 WIB
3. MPH - 00:10 WIB
-----
Masukkan nomor film yang ingin diedit: 1
Judul Film Baru: LLS
Jam Tayang Baru (HH:MM): 00:08
Jadwal berhasil diedit!
=====
- Admin ❤️
```

Gambar 14: Mengedit jadwal film berdasarkan jam tayang

```

=====
||      Waktu Saat Ini: 22:16 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 4
-----
=====
||      Daftar Film - Edit      ||
=====
1. Jumbo - 22:45 WIB
2. Keluarga Cemara - 23:00 WIB
3. Labubu - 23:00 WIB
4. Pandora - 22:30 WIB
-----
Masukkan nomor film yang ingin diedit: 3
Judul Film Baru: Labubu n Friends
Jam Tayang Baru (HH:MM): 23:30
Jadwal berhasil diedit!
=====
- Admin ❤️

```

Gambar 15: Mengedit jadwal film berdasarkan judul dan jam tayang

3.6 Pengguna memilih opsi 5 (Menghapus jadwal film)

```

=====
||      Waktu Saat Ini: 00:04 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 🍷
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar
-----
Pilih menu: 5
-----
=====
||      Daftar Film yang Tersedia      ||
=====
1. LLS - 00:08 WIB
2. MDR - 00:10 WIB
3. MIB - 00:15 WIB
-----
Masukkan nomor film yang ingin dihapus: 1
Film nomor 1 berhasil dihapus.
=====
- Admin ❤️

```

Gambar 16: Menghapus jadwal film dari daftar film

3.7 Pengguna memilih opsi 6 (Melihat history jadwal film)

```
=====
||      Waktu Saat Ini: 00:06 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👍
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 6

=====
||      Riwayat Film      ||
=====
Tanggal: 2025-06-12
Riwayat Film:
  1. LLS - 00:08 WIB
=====
Tanggal: 2025-06-11
Riwayat Film:
  1. Keluarga Cemara - 23:55 WIB
  2. Pandora - 23:56 WIB
=====
- Admin ❤️
```

Gambar 17: History jadwal film yang sudah lewat

3.8 Pengguna memilih opsi 0 (Keluar dari menu)

```
=====
||      Waktu Saat Ini: 00:07 WIB      ||
=====
||      MENU      BIOSKOP      ||
=====
1. Daftar Film 📄
2. Cari Jadwal Film
3. Tambah Jadwal Film
4. Edit Film 👍
5. Hapus Jadwal Film
6. Lihat History Film Lewat
0. Keluar

-----
Pilih menu: 0

=====
|| Terima kasih telah menggunakan program ini. ||
=====
```

Gambar 18: Tampilan saat keluar dari menu

4 Penjelasan

4.1 Mengimpor Library Python

- **import json** digunakan untuk mengimpor atau memanggil modul json bawaan Python. json (JavaScript Object Notation) adalah format penyimpanan data berbasis teks yang ringan dan mudah dibaca, dimana dalam program ini json berfungsi untuk menyimpan jadwal bioskop yang berisi judul dan jam tayang film, menyimpan history jadwal bioskop yang berisi film-film yang sudah lewat, dan membaca ulang file tersebut saat program dijalankan agar data tetap tersedia.
- **from datetime import datetime, time, timezone, timedelta, date** adalah instruksi untuk mengimpor fungsi dan class waktu tertentu dari modul datetime, yang digunakan untuk mengatur dan memanipulasi waktu dan tanggal, dalam program ini berfungsi untuk menentukan waktu saat ini dengan zona WIB, memvalidasi jam tayang (mengubah input jam string menjadi objek jam), membuat label tanggal pada history (menyimpan tanggal input dengan format "yyyy-mm-dd").

4.2 Data Structure Definitions

- **class JadwalFilm**, kelas ini mendefinisikan struktur data satu node untuk membentuk LinkedList, di mana setiap node menyimpan judul film, waktu tayang, dan `self.next` adalah pointer ke node berikutnya. `__init__` dalam kelas ini berfungsi sebagai konstruktor untuk menetapkan nilai awal judul, jam, dan pointer ke node berikutnya. Kelas ini digunakan untuk merepresentasikan setiap film yang terdaftar secara dinamis.
- **class LinkedList**, kelas LinkedList ini digunakan untuk mengelola daftar film yang ditambahkan oleh admin. Dengan menggunakan struktur data dinamis yaitu LinkedList, program ini akan berjalan lebih fleksibel dalam melakukan penambahan, penghapusan, dan penelusuran data. Kelas ini berisi beberapa method yaitu:
 - `def insert(self, judul, jam:time)`: digunakan untuk menambahkan node baru ke dalam LinkedList. Jika head kosong, node baru akan menjadi head. Jika sudah ada node, maka node baru ditambahkan ke bagian akhir (tail).
 - `def to_array(self)`: mengubah LinkedList menjadi array dalam bentuk pasangan (judul, jam). Hal ini bertujuan agar data dapat disortir dengan lebih mudah, disimpan ke file JSON, atau digunakan untuk operasi yang membutuhkan indeks.
 - `def from_array(self, data)`: kebalikan dari `to_array()`, berfungsi untuk menerima array dan menyusunnya kembali menjadi LinkedList. Fungsi ini biasa digunakan atau akan dipanggil setelah proses penyortiran (sorting).
 - `def sort_by(self, key='judul')`: fungsi ini menerima parameter key untuk menentukan apakah pengurutan berdasarkan judul atau jam. Fungsi ini menggunakan algoritma **insertion sort** dan melibatkan langkah-langkah:

- * Mengambil data dari LinkedList menggunakan `self.to_array()` agar mudah diurutkan.
 - * Melakukan perulangan dari indeks ke-1 hingga indeks terakhir pada array.
 - * Menyimpan elemen yang sedang diproses dalam variabel `key_item`.
 - * Inisialisasi `j = i - 1` sebagai indeks baru untuk dibandingkan dengan data pada indeks setelahnya (data pada indeks `i`).
 - * `while j >= 0 and (...)`: perulangan ke belakang jika syarat pengurutan masih terpenuhi.
 - * `(key == 'judul' and data[j][0].lower() > key_item[0].lower())`: pengecekan urutan berdasarkan judul secara alfabetik tidak sensitif huruf besar/kecil.
 - * `(key != 'judul' and data[j][1] > key_item[1])`: pengecekan urutan berdasarkan jam tayang.
 - * Jika memenuhi kondisi pada while (jika data pada indeks ke-`j` lebih dari data sebelah kanan), maka akan menjalankan perintah `data[j + 1] = data[j]`, yaitu menggeser elemen yang lebih besar ke kanan.
 - * `j -= 1`: menggeser indeks pembanding ke kiri.
 - * `data[j + 1] = key_item`: menempatkan `key_item` ke posisi yang tepat.
 - * Terakhir, `self.from_array(data)` dipanggil untuk mengembalikan data terurut ke dalam struktur data dinamis yaitu LinkedList.
- `def hapus_by_index_sorted_judul(self, index):` berguna untuk menampilkan daftar film yang sudah diurutkan berdasarkan judul, menghapus film berdasarkan input nomor dari user/admin, dan setelah dihapus, data tersebut tercatat ke dalam history jadwal film. fungsi tersebut berjalan dengan langkah sebagai berikut :
- * memanggil `data = self.to_array()` yaitu data dalam bentuk array (data statis)
 - * mengurutkan array berdasarkan judul
 - * menghapus film sesuai nomor yang diinput
 - * jika nomor yang di-input tidak valid, maka akan ada pesan warning dan akan menyuruh untuk menginput nomor yang valid
 - * menyimpan ulang hasilnya ke dalam struktur data dinamis (LinkedList)
 - * menyimpan film yang dihapus ke dalam file **history_jadwal_bioskop.json**.
- `def tampilkan_dengan_nomor(self, by='judul')`: fungsi ini menampilkan semua data film yang ada di dalam LinkedList dengan penomoran urut, yang akan berguna dalam beberapa fitur di program ini, misalnya dalam fitur edit jadwal yang akan menampilkan jadwal film beserta jamnya dalam penomoran terurut, fitur hapus jadwal film, dan fitur daftar film (untuk melihat semua data film yang ada). Jika data jadwal film tidak ada, maka program akan memberi pesan *"Daftar Kosong"*.
- **class Queue**, kelas Queue ini dirancang untuk mengelola antrian jadwal film bioskop. Beberapa fiturnya yaitu menambahkan jadwal film, menghapus jadwal yang sudah melewati waktu sekarang, dan menampilkan jadwal yang

masih aktif. Data jadwal yang sudah lewat akan disimpan ke dalam sebuah file riwayat/histori dalam format .json. Dalam Queue ini berisi beberapa method yaitu:

- `def __init__(self):`
Fungsi `__init__` adalah konstruktor dari kelas Queue. Fungsi ini dipanggil secara otomatis ketika sebuah objek dari kelas Queue dibuat. Tugas utamanya adalah menginisialisasi atribut `self.items` menjadi sebuah list kosong. Atribut `self.items` ini akan digunakan untuk menyimpan antrian jadwal film, di mana setiap elemen dalam list akan berupa tuple yang berisi judul film, waktu tayang, dan tanggal tayang. Dengan inisialisasi ini, setiap objek Queue yang baru dibuat akan memiliki antrian yang kosong.
- `def enqueue(self, judul, jam: time):`
Fungsi ini digunakan untuk menambahkan jadwal film baru ke dalam antrian. Fungsi ini menerima dua argumen: judul (judul film dalam bentuk string) dan jam (sebagai objek time yang merepresentasikan waktu tayang film). Dalam fungsi ini, program secara otomatis mengambil tanggal saat ini yaitu menggunakan `date.today().isoformat()` dan disimpan dalam variabel `tanggal_input`. Kemudian, sebuah tuple yang berisi (judul, jam, dan `tanggal_input`) ditambahkan ke akhir list `self.items` menggunakan method `append` (method bawaan dari Python). Dengan demikian, jadwal film yang baru ditambahkan akan berada di posisi terakhir dalam antrian, menunggu gilirannya untuk tayang atau diperiksa apakah sudah kadaluwarsa.
- `def dequeue_expired(self, current_time: time):`
fungsi `dequeue_expired` digunakan untuk menghapus jadwal film yang sudah kadaluarsa (expired) dari antrian dan memindahkannya ke dalam histori. Jadwal film dianggap kadaluarsa jika waktu tayangnya (jam) lebih kecil (telah melewati) atau sama dengan `current_time` (waktu saat ini). Fungsi ini juga memeriksa apakah jadwal film tersebut sudah tercatat dalam histori untuk tanggal yang sama. Fungsi tersebut berjalan dengan prosedur berikut:
 - * Pertama, fungsi ini menginisialisasi dua list kosong: `new_expired` untuk menyimpan jadwal film yang kadaluarsa dan belum ada di histori dan `remaining` untuk menyimpan jadwal film yang masih berlaku. Kemudian, fungsi ini mencoba membaca file JSON `"history_jadwal_bioskop.json"` untuk mendapatkan data histori jadwal film sebelumnya. Jika file tidak ditemukan (`FileNotFoundError`), maka histori diinisialisasi sebagai dictionary kosong. Jika terjadi kesalahan dalam membaca atau mengurai file JSON (`json.JSONDecodeError`), pesan error dicetak dan histori diinisialisasi sebagai dictionary kosong.
 - * Selanjutnya, fungsi ini melakukan iterasi melalui setiap item (tuple (judul, jam, tgl)) dalam `self.items`. Untuk setiap item, waktu tayang (jam) dikonversi menjadi string dengan format "HH:MM" menggunakan `jam.strftime('%H:%M')`. Kemudian, dilakukan pengecekan apakah jam kurang dari atau sama dengan `current_time`. Jika ya, maka jadwal film dianggap kadaluarsa. Selanjutnya, diperiksa

apakah tanggal (`tgl`) sudah ada sebagai key dalam dictionary histori jadwal film, dan jika ada, apakah tuple (`judul`, `jam_str`) sudah ada dalam list yang menjadi value dari key `tgl` tersebut. Jika jadwal film belum ada dalam history untuk tanggal tersebut, maka tuple (`judul`, `jam`, `tgl`) ditambahkan ke list `new_expired`. Jika jadwal film belum kadaluarsa, maka tuple (`judul`, `jam`, `tgl`) ditambahkan ke list `remaining`.

- * Setelah iterasi selesai, `self.items` diupdate dengan list `remaining` (jadwal film yang masih berlaku). Jika ada jadwal film yang kadaluarsa (`new_expired` tidak kosong), maka fungsi `simpan_ke_history` dipanggil untuk menyimpan jadwal film tersebut ke dalam file history. Terakhir, fungsi ini mengembalikan list `new_expired`.
- `def from_array(self, data):`
Fungsi `from_array` menyediakan cara yang cepat dan mudah untuk mengisi antrian dari sumber data eksternal, seperti sebuah list atau array Python. Fungsi ini pertama-tama akan mengosongkan antrian yang ada (`self.items = []`). Kemudian, ia akan mengulang melalui setiap item dalam data yang diberikan. Diasumsikan bahwa setiap item dalam data adalah tuple atau list yang berisi judul dan jam tayang. Untuk setiap pasangan (`judul`, `jam`) ini, fungsi `enqueue` akan dipanggil, sehingga setiap jadwal akan ditambahkan ke antrian dengan tanggal hari ini secara otomatis.

4.3 File JSON

Bagian ini digunakan untuk membaca dan menyimpan data dalam file dengan ekstensi **.json**.

- `def simpan_ke_json(nama_file, data)`
Fungsi tersebut digunakan untuk menyimpan data jadwal film kedalam file dengan format JSON, data jam tayang film diubah terlebih dahulu ke dalam format string "jam:menit" agar dapat disimpan dengan benar ke dalam file JSON. Proses penyimpanan dilakukan dengan membuka file menggunakan fungsi `with open(nama_file, 'w') as file` jam, kemudian data yang telah diubah disimpan ke dalam file tersebut menggunakan format JSON dengan menggunakan fungsi `json.dump`.
- `def baca_dari_json(nama_file)`
Fungsi tersebut digunakan untuk membaca file JSON yang berisi data jadwal film, kemudian mengubah data tersebut kembali menjadi judul film dan jam tayang dalam tipe waktu `datetime.time`. Proses ini diawali dengan membuka file menggunakan perintah `with open(nama_file, 'r') as f`, isi file yang dibaca dan disimpan ke dalam variabel data menggunakan fungsi `json.load(f)`. Selanjutnya, data yang sebelumnya disimpan dalam format string akan diubah kembali menjadi format waktu yang menggunakan `datetime.time`, jika file JSON tidak ditemukan maka program tidak error hanya mengembalikan list kosong.
- `def simpan_ke_history(nama_file, data_baru):`
Fungsi diatas digunakan untuk menyimpan riwayat jadwal film yang sudah melewati jam tayang ke dalam file JSON, ini dikelompokkan berdasarkan

tanggal penayangan serta untuk menghindari duplikasi data jam tayang film. Blok `try` digunakan untuk membaca isi file JSON yang sudah ada, namun jika file tersebut belum tersedia maka akan dibuat data lama sebagai dictionary kosong. Perulangan `for` digunakan untuk memproses setiap data baru yang berisi informasi judul, jam tayang dan tanggal, kemudian pada bagian `if tanggal not in data_time`, program memeriksa apakah tanggal tersebut sudah tersedia lalu membuat daftar kosong untuk tanggal jika belum tersedia. `jam_str` berfungsi untuk mengubah objek waktu menjadi string dalam format "HH:MM" yang kemudian digunakan untuk menambahkan data baru ke dalam daftar. `With open(nama_file, 'w') as f:` digunakan untuk menyimpan data riwayat yang telah diperbarui ke dalam file JSON.

- `def update_history`
Fungsi diatas digunakan untuk memperbarui data jadwal film dengan menghapus jadwal film yang sudah melewati jam tayang, kemudian menyimpan data baru ke dalam file `jadwal_bioskop.json`. Bagian `waktu_sekarang` berfungsi untuk mendapatkan waktu saat ini dalam zona waktu GMT+7. Fungsi `dequeue_expired` digunakan untuk menghapus data film yang sudah kadaluwarsa dan data tersebut disimpan dalam variabel `expired_data`. Selanjutnya `linkedlist` digunakan untuk membuat struktur data berisi judul dan jam dari setiap film yang masih berlaku. Data pada `linkedlist` diubah menjadi array dan disimpan kembali ke file JSON agar tetap tersimpan saat program dijalankan ulang.
- `def baca_history_dengan_tanggal(nama_file)`
Fungsi diatas digunakan untuk membaca file JSON yang berisi data jadwal film beserta tanggalnya. Pada blok `try` program membuka file dengan nama yang diberikan, jika berhasil isi file dibaca menggunakan `json.load`. Namun, jika file tidak ditemukan maka program tidak error hanya mengembalikan list kosong.
- `def validasi_waktu(jam):`
Fungsi diatas digunakan untuk mengecek apakah format waktu yang dimasukkan sesuai dan berada pada rentang waktu yang valid (HH:MM). Pada blok `try`, string pada jam dipisahkan menggunakan titik dua (`:`) lalu diubah menjadi bilangan bulat menggunakan `map(int, jam.split(':'))`. `If` digunakan untuk memastikan bahwa jam berada diantara 0 – 23 dan menit antara 0 – 59. Jika format salah atau nilai berada diluar rentang, `except` akan menangani kesalahan dan mengembalikan bahwa input tidak valid.

4.4 Main Program

Bagian main program berfungsi sebagai pusat kendali dari seluruh proses manajemen jadwal film yang ditayangkan di bioskop. Program dimulai dengan membuat variabel `linkedlist` dan `antrian`. Variabel `linkedlist` digunakan untuk menyimpan daftar film yang akan ditayangkan dalam bentuk struktur data `linked list`, sementara `antrian` digunakan untuk menyimpan daftar film yang dijadwalkan secara berurutan dalam bentuk `queue`, sehingga jika terdapat film yang sudah melewati jam tayangnya maka film tersebut akan keluar dari `antrian`.

Setelah itu program membaca data awal dari file `jadwal_bioskop.json` menggunakan fungsi `baca_dari_json`. File ini berisi data film yang telah disimpan

sebelumnya dengan format JSON, dan setiap datanya berupa pasangan judul dan jam tayang. Data yang sudah dibaca dimasukkan satu per satu ke dalam linkedlist dengan metode insert.

Setelah semua data dimasukkan, program akan melakukan sortir berdasarkan jam tayang agar daftar film yang ditampilkan berdasarkan urutan jadwal penayangan. Proses sortir dilakukan dengan metode `sort_by('jam')` dan hasil sortir disusun ulang ke dalam linked list menggunakan metode `from_array`. Hasil dari linked list yang telah diurutkan juga dimasukkan ke dalam queue antrian dengan tujuan agar film yang sudah melewati jam tayang bisa dikeluarkan menggunakan prinsip FIFO (First-In First-Out).

Program kemudian mengecek waktu saat ini dengan menggunakan waktu lokal (WIB) menggunakan `datetime.now(timedelta(hours=7))`. Dengan waktu ini, program menggunakan metode `dequeue_expired` untuk menghapus film-film yang jam tayangnya sudah lewat dari queue dan mencatatnya ke dalam file `history_jadwal_bioskop.json`. Setelah data lama (kedaluwarsa) dikeluarkan, isi linkedlist diperbarui kembali berdasarkan data dalam queue dan hasil pembaruan disimpan ulang ke file `jadwal_bioskop.json`. Hal ini dilakukan agar data film yang ditampilkan selalu valid dan up to date dengan waktu sekarang.

Setelah itu, program menampilkan waktu saat ini ke user dalam format HH:MM. Dengan menggunakan perulangan tak hingga (`while True`), program akan terus berjalan dan menampilkan ulang menu utama setiap proses selesai. Adapun pilihan menu yang terdiri dari menampilkan daftar film (1), mencari film (2), menambah jadwal film (3), mengedit jadwal film (4), menghapus jadwal film (5), melihat history film yang sudah tayang (6), dan keluar dari program (0). Fungsi `menu()` dipanggil untuk menampilkan tampilan menu utama, kemudian pengguna diminta untuk memasukkan pilihan melalui `input("Pilih menu: ")`. Setelah itu fungsi `update_history()` dijalankan untuk memperbarui informasi jadwal film, seperti menghapus data film yang sudah lewat waktu tayangnya, agar data tetap up to date.

1. Ketika pengguna memilih menu **1**, program akan menampilkan tampilan untuk melihat daftar film. Selanjutnya, pengguna memiliki dua pilihan untuk menampilkan daftar film sesuai urutan: berdasarkan **judul** atau **waktu tayang**. Program meminta input berupa angka 1 (berdasarkan judul) atau 2 (berdasarkan waktu tayang), dan validasi dilakukan untuk memastikan input sesuai dengan pilihan yang ada. Apabila input yang diberikan tidak valid (misal bukan angka 1 atau 2), program akan memberi pesan bahwa input tidak valid dan meminta pengguna untuk mengulangi input hingga benar.

- Jika pengguna memilih **opsi 1**, program akan menyusun daftar film berdasarkan judul secara alfabetis menggunakan `linkedlist.sort_by('judul')`. Setelah pengurutan data, daftar film ditampilkan menggunakan `linkedlist.tampilkan_dengan_nomor('judul')`, di mana setiap daftar film yang telah diurutkan secara alfabetis disajikan dengan nomor urut.
- Jika pengguna memilih **opsi 2**, daftar film diatur berdasarkan urutan waktu tayang menggunakan `linkedlist.sort_by('jam')`. Setelah proses pengurutan selesai, daftar film ditampilkan menggunakan `linkedlist.tampilkan_dengan_nomor('jam')`, sehingga pengguna dapat melihat jadwal film secara berurutan sesuai jam tayang.

Validasi input menjamin bahwa hanya nilai yang valid (1 atau 2) yang dapat diterima, sehingga mengurangi kemungkinan terjadinya kesalahan. Program ini memberikan kebebasan kepada pengguna untuk menampilkan data yang penting sesuai dengan preferensi, baik berdasarkan judul maupun waktu pemutaran film.

2. Ketika pengguna memilih menu **2**, program akan menampilkan tampilan untuk mencari jadwal film. Selanjutnya, program memberikan dua opsi pencarian: opsi 1 (berdasarkan **judul**) atau opsi 2 (berdasarkan **waktu tayang**).

Dalam pencarian berdasarkan **judul**, pengguna dapat memasukkan kata kunci, baik sebagian maupun lengkap dari judul film karena pencarian ini dilakukan menggunakan partial search, yaitu metode pencarian yang mencocokkan sebagian dari kata kunci dengan data yang tersedia. Hal tersebut memungkinkan pengguna untuk menemukan data meskipun hanya sebagian kata yang dimasukkan. Program kemudian memeriksa semua data film yang tersimpan dalam `linkedlist`, yang sebelumnya diubah menjadi array dengan `to_array()`.

Proses pencarian dilakukan tanpa memperhatikan huruf besar dan kecil, karena antara kata kunci dan data yang tersedia disetel dalam lowercase atau non-kapital dengan `.lower()`, untuk memastikan kata kunci dapat terdeteksi tanpa memperdulikan perbedaan huruf kapital atau non-kapital. Apabila terdapat kesesuaian, program akan menunjukkan daftar film yang berkaitan, dilengkapi dengan nomor urut dan waktu tayang dalam format HH:MM WIB. Sebaliknya, jika tidak ada film yang cocok, program akan menginformasikan bahwa *"Film tidak ditemukan."*

Dalam pencarian berdasarkan **waktu tayang**, pengguna diminta untuk menginput waktu dalam format HH:MM, yang selanjutnya divalidasi dengan mengonversinya menjadi objek waktu melalui `datetime.strptime()`. Apabila waktu input valid, yaitu dari 00:00-23:59, program akan membandingkan dengan jadwal tayang setiap film yang tersimpan. Jika terdapat film dengan jadwal tayang yang sesuai, hasil pencarian akan ditampilkan dalam format yang sama seperti pencarian yang dilakukan berdasarkan judul. Jika waktu yang dimasukkan valid, tetapi tidak ada film yang tersedia pada waktu tersebut, program akan menampilkan pesan kesalahan bahwa *"Film tidak ditemukan"*. Namun, jika waktu yang dimasukkan tidak valid, maka program akan menampilkan pesan kesalahan *"Format jam tidak valid"*.

3. Ketika user memilih menu **3**, program akan menampilkan tampilan untuk menambah jadwal film. Program akan meminta input berupa judul film yang ingin ditambahkan dan jam tayang dalam format HH:MM. Input jam tayang kemudian divalidasi menggunakan fungsi `validasi_waktu(jam_input)`, yang bertujuan untuk memeriksa apakah format jam yang dimasukkan sesuai.

Jika input jam tayang tidak valid, maka akan ditampilkan pesan warning *"Tolong masukkan waktu yang valid."* dan program akan meminta pengguna untuk terus memasukkan input jam tayang yang valid. Jika format waktu valid, maka input-an jam tayang tersebut akan dikonversi menggunakan `datetime.strptime(jam_input, '%H:%M').time()`.

Selanjutnya, program mengambil waktu saat ini menggunakan `datetime.now(timezone(timedelta(hours=7))).time()`. Setelah itu, dilakukan pengecekan apakah jam tayang yang dimasukkan kurang dari dari

waktu sekarang, yang berarti jadwal sudah lewat. Jika demikian, program akan menampilkan pesan *“Jadwal tidak bisa ditambahkan karena waktu sudah lewat.”* dan kembali ke menu awal.

Jika semua pengecekan lolos, maka program melanjutkan dengan menambahkan data ke dalam struktur `linkedlist` menggunakan `linkedlist.insert(judul, jam)`. Setelah ditambahkan, data diurutkan berdasarkan jam tayang menggunakan `linkedlist.sort_by('jam')`. Hasil `linkedlist` yang sudah terurut kemudian dikonversi menjadi array melalui `linkedlist.to_array()`, dan hasil array ini digunakan untuk memperbarui antrian (`queue`) melalui `antrian.from_array(data)`. Setelah itu, data jadwal disimpan secara permanen ke dalam file `jadwal_bioskop.json` dengan menggunakan fungsi `simpan_ke_json(jadwal_bioskop.json)`, sehingga informasi tidak hilang ketika program ditutup. Terakhir, ditampilkan pesan kepada pengguna bahwa jadwal berhasil ditambahkan.

4. Ketika memilih menu **4**, user mampu mengedit data film yang sudah ada. Pertama memanggil `linkedlist.sort_by('judul')` untuk menyusun daftar film yang telah tersedia berdasarkan judul. Setelah itu, data dalam `linkedlist` dikonversi menjadi array dan ditampilkan satu per satu melalui perulangan `for`, lengkap dengan indeks dan jam tayang yang telah diformat HH:MM dengan `strftime('%H:%M')`. Sehingga, program menampilkan daftar film yang telah tersedia berdasarkan urutan alfabetis dengan nomor urutannya. Hal ini bertujuan agar pengguna bisa memilih nomor film yang ingin diedit.

Program kemudian meminta input dari pengguna berupa nomor film yang ingin diedit. Input divalidasi apakah berada dalam rentang yang sesuai dengan jumlah data film yang tersedia. Jika nomor film yang dipilih valid, maka pengguna diminta memasukkan judul film baru dan jam tayang baru. Jam tayang yang dimasukkan divalidasi dengan fungsi `validasi_waktu()` untuk memastikan format input adalah HH:MM. Jika input jam tidak valid, maka akan ditampilkan pesan warning *“Tolong masukkan waktu yang valid.”* dan program akan terus meminta pengguna untuk memasukkan input jam yang valid.

Jika format waktu valid, maka string waktu diubah menjadi objek waktu menggunakan `datetime.strptime()`, lalu dibandingkan dengan waktu saat ini. Jika waktu tayang yang baru telah lewat dari waktu sekarang, maka jadwal tidak bisa diubah, dan pengguna mendapat peringatan *“Tidak dapat mengganti dengan jam yang sudah lewat.”*. Namun, jika waktu valid dan belum lewat, data yang lama dalam array diganti dengan tuple (`judul_baru, jam_baru`) pada indeks yang sesuai.

Setelah data diperbarui di array, `linkedlist.from_array(data)` digunakan untuk menyinkronkan kembali isi `linkedlist`. Kemudian daftar kembali disortir berdasarkan jam tayang. Selanjutnya, antrian diperbarui berdasarkan isi `linkedlist` terbaru dengan `antrian.from_array()`, sehingga antrian tetap konsisten dengan data film yang baru diedit. Terakhir, semua data disimpan kembali ke dalam file JSON (`jadwal_bioskop.json`) menggunakan fungsi `simpan_ke_json()` dan program memberikan konfirmasi berupa pesan *“Jadwal berhasil diedit!”* kepada pengguna.

Jika pengguna memasukkan nomor film yang tidak valid (di luar jangkauan), maka program akan menampilkan pesan *“Nomor tidak valid.”*. Jika pengguna

memasukkan input yang tidak bisa dikonversi menjadi integer (misalnya huruf atau simbol), maka blok `except ValueError` akan menangkap kesalahan tersebut dan mencetak *"Input tidak valid."*. Secara keseluruhan, bagian kode ini memastikan proses pengeditan berjalan dengan aman dan efisien.

5. Ketika memilih menu 5, pengguna mampu menghapus film. Pertama, `linkedlist` diurutkan berdasarkan judul film agar daftar tersaji secara alfabetis. Kemudian, data diubah menjadi array dengan `to_array()` dan ditampilkan ke pengguna lengkap dengan nomor urut, judul, dan jam tayangnya. Sehingga, muncul tampilan daftar film yang tersedia beserta urutannya secara alfabetis.

Setelah itu, pengguna diminta memasukkan nomor film yang ingin dihapus. Input ini diproses dalam blok `try` untuk menangani error jika input bukan berupa angka. Jika berupa angka, metode `hapus_by_index_sorted_judul(index)` dipanggil untuk menghapus data pada indeks tersebut yang sudah disortir berdasarkan judul. Dalam metode tersebut, jika angka yang dimasukkan tidak valid (di luar jangkauan urutan daftar film), maka program akan menampilkan pesan *"Tolong masukkan nomor yang valid."* dan meminta pengguna untuk terus memasukkan angka yang valid. Setelah itu, Hasilnya disimpan dalam `new_data`.

Langkah selanjutnya adalah memperbarui isi `linkedlist` berdasarkan data terbaru (`new_data`) dengan memanggil `from_array()`, lalu menyortir ulang berdasarkan jam tayang menggunakan `sort_by('jam')`. Setelah itu, data yang baru, dikonversi ke dalam struktur `Queue` melalui `antrian.from_array(new_data)` agar sistem antrian tetap selaras dengan isi jadwal terkini.

Terakhir, seluruh data baru disimpan kembali ke dalam file JSON bernama `jadwal_bioskop.json` menggunakan fungsi `simpan_ke_json()`, dan file ini menjadi bentuk penyimpanan permanen dari jadwal yang telah diperbarui. Bila terjadi kesalahan input (seperti memasukkan karakter non-angka), maka blok `except` akan menampilkan pesan *"Input tidak valid."* sebagai bentuk penanganan error yang ramah pengguna.

6. Ketika memilih menu **6**, pengguna dapat melihat daftar film yang sudah terlewat (histori film) sesuai dari file JSON. Proses ini dimulai dengan membuka file `history_jadwal_bioskop.json` menggunakan fungsi `baca_history_dengan_tanggal()`. Fungsi ini menghasilkan data dalam bentuk *dictionary* yang memuat daftar film yang dikelompokkan menurut tanggal.

Setelah data diperoleh, program mengecek apakah ada data history yang tersedia. Apabila tidak tersedia, akan ditampilkan pesan *"Belum ada history."*. Akan tetapi, jika data ada, program akan menunjukkan daftar riwayat dengan susunan terstruktur dari tanggal yang terbaru hingga yang terlama. Data ini disusun dengan fungsi `sorted()` dengan parameter `reverse=True`.

Setiap tanggal, program menampilkan *header* tanggal diikuti dengan riwayat daftar film yang terdaftar pada hari tersebut. Daftar ini disajikan dalam format nomor urut, judul film, dan waktu tayang dalam format HH:MM WIB. Waktu tayang diambil langsung dari data JSON, dan setiap entri film ditampilkan melalui iterasi dengan menggunakan `enumerate()`.

7. Saat memilih menu **0**, pengguna akan keluar dari aplikasi. Ketika pengguna memilih opsi ini berarti pengguna ingin menghentikan program. Kode terse-

but akan mencetak ucapan terima kasih yang menunjukkan pesan utama *"Terima kasih telah menggunakan program ini."* sebagai penutup dari program.

Setelah mencetak pesan tersebut, program menjalankan perintah `break`. Perintah ini digunakan untuk menghentikan proses *loop* utama, yang memakai struktur `while True:`. Dengan perintah ini, aplikasi akan keluar dari *loop* dan semua proses akan berhenti, menunjukkan bahwa pengguna telah selesai menggunakan program.

Akan tetapi, jika pengguna memberikan input yang bukan angka yang ada dalam pilihan menu, maka bagian `else` akan dijalankan. Blok ini akan menunjukkan pesan *"PILIHAN TIDAK VALID"* untuk memberi tahu pengguna bahwa input yang dimasukkan tidak cocok dengan opsi menu yang ada.

Setelah setiap menjalankan blok `if`, `elif` atau `else`, sesuai dengan pilihan menu, program menampilkan tulisan *"-Admin"* sebagai identifikasi dari sistem. Ini berfungsi sebagai penutup untuk setiap siklus proses input dan memberikan kesan yang bersahabat kepada pengguna.

Pustaka

- [1] Romadhonia, R. W. (2025), *Struktur Data dan Algoritma T.A. 2024/2025*.
- [2] Contentstack, "Partial Search", *Contentstack Docs*, <https://www.contentstack.com/docs/content-managers/search-content/partial-search>.
- [3] Hostinger, "Apa Itu JSON? Pengertian, Fungsi, dan Cara Menggunakannya", *Hostinger Tutorial*, <https://www.hostinger.com/id/tutorial/apa-itu-json>.

Lampiran

PPT: <https://www.overleaf.com/1712244531tgkshmnnygtsb#f3f7ab>

Laporan: <https://www.overleaf.com/8424293482snqbhfckccmx#5178a5>