# Lab 2 - Bangun Indeks & Cari

ukuran chunk 1000, overlap 15%, backend: neural

## Langkah 1. Membangun indeks

6 PDF berada di folder dokumen/. Setiap file melaporkan jumlah chunk-nya:

| File | Jumlah chunk |
|---|---|
| 1_Pengenalan_Kecerdasan_Buatan.pdf | 4 |
| 2_Machine_Learning_dan_Deep_Learning.pdf | 4 |
| 3_Etika_Risiko_dan_Masa_Depan_AI.pdf | 4 |
| etika_keamanan_ai.pdf | 4 |
| machine_learning_dasar.pdf | 4 |
| pengantar_rag.pdf | 5 |
| Total | 25 |

```
1_Pengenalan_Kecerdasan_Buatan.pdf: 4 chunk
2_Machine_Learning_dan_Deep_Learning.pdf: 4 chunk
3_Etika_Risiko_dan_Masa_Depan_AI.pdf: 4 chunk
etika_keamanan_ai.pdf: 4 chunk
machine_learning_dasar.pdf: 4 chunk
pengantar_rag.pdf: 5 chunk
Total 25 chunk | ukuran=1000 overlap=15% | backend=neural
```

## Langkah 2. Lima pertanyaan dan pemeriksaan chunk teratas

Sebuah chunk dianggap relevan bila memuat semua kata kunci jawaban (huruf besar/kecil dan pemisah baris diabaikan). Tabel menunjukkan 3 chunk teratas.

### Siapa yang memperkenalkan istilah kecerdasan buatan dan pada tahun berapa?

| Peringkat | Skor | Sumber | Relevan? | Cuplikan |
|---|---|---|---|---|
| 1 | 0.708 | 1_Pengenalan_Kecerdasan_Buatan.pdf#0 | ya | Pengenalan Kecerdasan Buatan Konsep dasar, sejarah, dan jenis-jenis AI 1. Apa Itu Kecerdasan Buatan? Kecerdasan Buatan atau Artificial Intel... |
| 2 | 0.594 | 1_Pengenalan_Kecerdasan_Buatan.pdf#2 | tidak | erdasarkan cara kerjanya, AI juga dibedakan menjadi mesin reaktif, mesin dengan memori terbatas, serta konsep teoretis seperti theory of min... |
| 3 | 0.553 | 3_Etika_Risiko_dan_Masa_Depan_AI.pdf#0 | tidak | Etika, Risiko, dan Masa Depan AI Menggunakan kecerdasan buatan secara bertanggung jawab 1. Mengapa Etika AI Penting? Karena AI semakin berpe... |

### Siapa juara Go yang dikalahkan oleh AlphaGo?

| Peringkat | Skor | Sumber | Relevan? | Cuplikan |
|---|---|---|---|---|
| 1 | 0.424 | 1_Pengenalan_Kecerdasan_Buatan.pdf#1 | ya | - 1980-an AI Winter: pendanaan menurun karena ekspektasi yang belum terpenuhi; sistem pakar populer. 1997 Deep Blue (IBM) mengalahkan juara ... |
| 2 | 0.154 | machine_learning_dasar.pdf#3 | tidak | erlalu besar membuat pelatihan divergen, sedangkan yang terlalu kecil membuat pelatihan sangat lambat. Pada mini-batch gradient descent, gra... |
| 3 | 0.140 | machine_learning_dasar.pdf#2 | tidak | ediksi positif yang benar, sedangkan recall mengukur seberapa banyak data positif yang berhasil ditemukan. F1-score adalah rata-rata harmoni... |

### Apa itu overfitting dan apa cirinya?

| Peringkat | Skor | Sumber | Relevan? | Cuplikan |
|---|---|---|---|---|
| 1 | 0.561 | machine_learning_dasar.pdf#1 | ya | i validasi, misalnya 72 persen. Model seolah menghafal data latih, termasuk derau di dalamnya. Underfitting terjadi ketika model terlalu sed... |
| 2 | 0.535 | 2_Machine_Learning_dan_Deep_Learning.pdf#2 | ya | bahasa besar; unggul pada teks, gambar, dan audio. GAN dan Diffusion Pembuatan gambar dan konten sintetis. 5. Tantangan dalam Pelatihan Mode... |
| 3 | 0.317 | machine_learning_dasar.pdf#2 | tidak | ediksi positif yang benar, sedangkan recall mengukur seberapa banyak data positif yang berhasil ditemukan. F1-score adalah rata-rata harmoni... |

### Arsitektur jaringan saraf apa yang dipakai untuk mengolah gambar?

| Peringkat | Skor | Sumber | Relevan? | Cuplikan |
|---|---|---|---|---|
| 1 | 0.455 | 1_Pengenalan_Kecerdasan_Buatan.pdf#3 | tidak | i Navigasi, optimasi rute, kendaraan otonom. Keuangan Deteksi penipuan, penilaian kredit, analisis risiko. Pertanian Pemantauan tanaman deng... |
| 2 | 0.420 | 1_Pengenalan_Kecerdasan_Buatan.pdf#2 | tidak | erdasarkan cara kerjanya, AI juga dibedakan menjadi mesin reaktif, mesin dengan memori terbatas, serta konsep teoretis seperti theory of min... |
| 3 | 0.373 | 3_Etika_Risiko_dan_Masa_Depan_AI.pdf#1 | tidak | an tentang penggunaan karya berhak cipta sebagai data pelatihan. 3. Dampak terhadap Pekerjaan AI berpotensi mengotomatisasi tugas-tugas ruti... |

### Apa saja isu etika utama dalam penggunaan AI?

| Peringkat | Skor | Sumber | Relevan? | Cuplikan |
|---|---|---|---|---|
| 1 | 0.874 | 3_Etika_Risiko_dan_Masa_Depan_AI.pdf#0 | ya | Etika, Risiko, dan Masa Depan AI Menggunakan kecerdasan buatan secara bertanggung jawab 1. Mengapa Etika AI Penting? Karena AI semakin berpe... |
| 2 | 0.770 | 3_Etika_Risiko_dan_Masa_Depan_AI.pdf#3 | tidak | di bidang sains, kesehatan, dan pendidikan. (cid:127) Perdebatan berkelanjutan tentang AGI dan keselamatan AI. 7. Kesimpulan Masa depan AI d... |
| 3 | 0.744 | 1_Pengenalan_Kecerdasan_Buatan.pdf#3 | tidak | i Navigasi, optimasi rute, kendaraan otonom. Keuangan Deteksi penipuan, penilaian kredit, analisis risiko. Pertanian Pemantauan tanaman deng... |

Ringkasan: chunk peringkat 1 relevan pada 4/5 pertanyaan; ada chunk relevan di 3 teratas pada 4/5.

## Langkah 3. Pertanyaan berparafrase

Dua pertanyaan memakai kata yang berbeda dari teks dokumen. Dibandingkan: pencarian semantik (neural) vs TF-IDF.

| Pertanyaan | Catatan | neural: top1 / top3 | TF-IDF: top1 / top3 |
|---|---|---|---|
| Mengapa model bahasa kadang memberi jawaban yang terdengar meyakinkan padahal salah? | Teks memakai kata 'halusinasi'; pertanyaan memakai 'terdengar meyakinkan padahal salah'. | tidak / ya | ya / ya |

| Pertanyaan | Catatan | neural: top1 / top3 | TF-IDF: top1 / top3 |
|---|---|---|---|
| Metode belajar mesin lewat coba-coba dengan imbalan dan sanksi itu namanya apa? | Teks memakai 'percobaan', 'hadiah', 'hukuman'; pertanyaan memakai 'coba-coba', 'imbalan', 'sanksi'. | tidak / ya | ya / ya |

Chunk teratas untuk "Mengapa model bahasa kadang memberi jawaban yang terdengar meyakinkan padahal salah?": 2_Machine_Learning_dan_Deep_Learning.pdf#2 (skor 0.501)

```
bahasa besar; unggul pada teks, gambar, dan audio. GAN dan Diffusion Pembuatan gambar dan konten
    sintetis. 5. Tantangan dalam Pelatihan Model (cid:127) Overfitting: model terlalu hafal data latih
    sehingga buruk pada data baru. (cid:127) Underfitting: model terlalu sederhana untuk menangkap pola.
    (cid:127) Kualitas data: data yang bias, kotor, atau tidak seimbang menurunkan kualitas model.
    (cid:127...
```

Chunk teratas untuk "Metode belajar mesin lewat coba-coba dengan imbalan dan sanksi itu namanya apa?": machine_learning_dasar.pdf#0 (skor 0.562)

```
Dasar-Dasar Machine Learning 1. Supervised Learning Pada supervised learning, model belajar dari data
    yang sudah berlabel. Tugas klasifikasi memprediksi kategori, misalnya menandai surel sebagai spam
    atau bukan spam. Tugas regresi memprediksi nilai kontinu, misalnya harga rumah berdasarkan luas dan
    lokasi. 2. Pembagian Data Data biasanya dibagi menjadi data latih 70 persen, data validasi 15
    persen...
```

## Kesimpulan

Pada 5 pertanyaan langsung, chunk peringkat 1 relevan pada 4/5 pertanyaan dan salah satu dari 3 chunk teratas relevan pada 4/5. Pada 2 pertanyaan parafrase, backend 'neural' menemukan chunk relevan di peringkat 1 pada 0/2 dan di 3 teratas pada 2/2, sedangkan pembanding TF-IDF 2/2 dan 2/2. Untuk "Arsitektur jaringan saraf apa yang dipakai untuk mengolah gambar?", chunk relevan pertama baru muncul di peringkat 6 dari 25 chunk. Pada parafrase ini, TF-IDF justru sama baik atau lebih baik daripada pencarian semantik. Korpus hanya 25 chunk dari 6 file, sehingga satu pertanyaan mengubah persentase cukup besar.

# Lampiran: kode lab2_indeks_cari.py

```
"""Lab 2 - Bangun indeks & cari (25 menit).

1. Letakkan 2-3 PDF di folder dokumen/, jalankan indeks, pastikan tiap file melaporkan jumlah chunk.
2. Ajukan 5 pertanyaan dengan query(); periksa apakah chunk teratas benar-benar relevan.
3. Ajukan 2 pertanyaan berparafrase; apakah pencarian semantik tetap menemukannya?

Versi ini memakai ketiga PDF tentang kecerdasan buatan:
    1_Pengenalan_Kecerdasan_Buatan.pdf
    2_Machine_Learning_dan_Deep_Learning.pdf
    3_Etika_Risiko_dan_Masa_Depan_AI.pdf
Jika belum ada di folder dokumen/, skrip menyalinnya dari folder skrip atau folder kerja.
"""
import io
import os
import re
import shutil
import sys
from contextlib import redirect_stdout

sys.path.insert(0, os.path.dirname(os.path.abspath(__file__)))
from laporan_utils import Laporan
from rag_utils import FOLDER_DOKUMEN, bangun_indeks, query

BASE = os.path.dirname(os.path.abspath(__file__))
UKURAN, OVERLAP = 1000, 0.15

DAFTAR_PDF = [
    "1_Pengenalan_Kecerdasan_Buatan.pdf",
    "2_Machine_Learning_dan_Deep_Learning.pdf",
    "3_Etika_Risiko_dan_Masa_Depan_AI.pdf",
]

# kw  = kata kunci yang HARUS ada (semuanya) di sebuah chunk agar dianggap relevan
# file = PDF tempat jawaban sebenarnya berada (dipakai untuk memeriksa kunci jawaban)
PERTANYAAN = [
    {"q": "Siapa yang memperkenalkan istilah kecerdasan buatan dan pada tahun berapa?",
     "kw": ["McCarthy", "1956"], "file": DAFTAR_PDF[0]},
    {"q": "Siapa juara Go yang dikalahkan oleh AlphaGo?",
     "kw": ["AlphaGo", "Lee Sedol"], "file": DAFTAR_PDF[0]},
    {"q": "Apa itu overfitting dan apa cirinya?",
     "kw": ["overfitting", "data latih"], "file": DAFTAR_PDF[1]},
    {"q": "Arsitektur jaringan saraf apa yang dipakai untuk mengolah gambar?",
     "kw": ["CNN", "Convolutional"], "file": DAFTAR_PDF[1]},
    {"q": "Apa saja isu etika utama dalam penggunaan AI?",
     "kw": ["Bias dan diskriminasi", "Privasi"], "file": DAFTAR_PDF[2]},
]
PARAFRASE = [
    {"q": "Mengapa model bahasa kadang memberi jawaban yang terdengar meyakinkan padahal salah?",
     "kw": ["halusinasi"], "file": DAFTAR_PDF[2],
     "catatan": "Teks memakai kata 'halusinasi'; pertanyaan memakai 'terdengar meyakinkan padahal
    salah'."},
    {"q": "Metode belajar mesin lewat coba-coba dengan imbalan dan sanksi itu namanya apa?",
     "kw": ["Reinforcement"], "file": DAFTAR_PDF[1],
     "catatan": "Teks memakai 'percobaan', 'hadiah', 'hukuman'; pertanyaan memakai 'coba-coba',
    'imbalan', 'sanksi'."},
]


# ---------------------------------------------------------------------------- utilitas

def siapkan_dokumen():
    """Pastikan ketiga PDF ada di FOLDER_DOKUMEN; salin dari folder skrip/kerja bila belum ada."""
    os.makedirs(FOLDER_DOKUMEN, exist_ok=True)
    for nama in DAFTAR_PDF:
        tujuan = os.path.join(FOLDER_DOKUMEN, nama)
        if os.path.isfile(tujuan):
            continue
        for d in (BASE, os.getcwd()):
            asal = os.path.join(d, nama)
            if os.path.isfile(asal):
                shutil.copy(asal, tujuan)
                print(f"  disalin: {nama} -> {FOLDER_DOKUMEN}")
                break
        else:
            print(f"  [PERINGATAN] {nama} tidak ditemukan di {FOLDER_DOKUMEN}, {BASE}, maupun
    {os.getcwd()}")


def rapikan(t):
    """Buang baris footer 'Halaman N' dan ratakan spasi/baris baru agar kata kunci lintas baris tetap
```

```
    cocok."""
    t = re.sub(r"^[ \t]*Halaman \d+[ \t]*$", " ", t, flags=re.MULTILINE)
    return " ".join(t.split()).lower()


def cocok_kata_kunci(teks, kw):
    """True bila SEMUA kata kunci ada di teks (tanpa peduli huruf besar/kecil dan pemisah baris)."""
    t = rapikan(teks)
    return all(k.lower() in t for k in kw)


def ringkas(t, n=150):
    t = " ".join(t.split())
    return t if len(t) <= n else t[:n] + "..."


def ya(x):
    return "ya" if x else "tidak"


def sumber(a):
    return f"{a['file']}#{a['no']}" if a else "-"


def jalankan(indeks, daftar, k=3):
    hasil = []
    for s in daftar:
        atas = query(indeks, s["q"], k)
        rel = [cocok_kata_kunci(a["teks"], s["kw"]) for a in atas]
        hasil.append((s, atas, rel))
    return hasil


def top1(rel):
    return bool(rel) and rel[0]


def periksa_kunci(idx, daftar):
    """Validasi soal: berapa chunk di indeks yang memuat semua kata kunci, dan di file mana?
    Soal yang kuncinya tidak ada di chunk mana pun tidak mungkin dijawab, jadi harus ketahuan."""
    hasil = []
    for s in daftar:
        ada = [i for i, c in enumerate(idx.chunks) if cocok_kata_kunci(c, s["kw"])]
        file_ada = sorted({idx.meta[i]["file"] for i in ada})
        hasil.append((s, len(ada), file_ada))
    return hasil


def peringkat_relevan(idx, s):
    """Peringkat chunk relevan pertama di SELURUH indeks (None bila tidak ada)."""
    semua = query(idx, s["q"], len(idx.chunks))
    for r, a in enumerate(semua, 1):
        if cocok_kata_kunci(a["teks"], s["kw"]):
            return r
    return None


# ---------------------------------------------------------------------------- utama

def main():
    # ---- Langkah 1
    print("=== Langkah 1: membangun indeks ===")
    siapkan_dokumen()
    buf = io.StringIO()
    with redirect_stdout(buf):
        idx = bangun_indeks(ukuran=UKURAN, overlap=OVERLAP, backend="auto")
    log_indeks = buf.getvalue()
    print(log_indeks)
    idx_leksikal = bangun_indeks(ukuran=UKURAN, overlap=OVERLAP, backend="tfidf", verbose=False)

    per_file = {}
    for m in idx.meta:
        per_file[m["file"]] = per_file.get(m["file"], 0) + 1
    hilang = [f for f in DAFTAR_PDF if per_file.get(f, 0) == 0]
    for f, n in per_file.items():
        print(f"  {f}: {n} chunk")
    if hilang:
        print("  [PERINGATAN] tidak ada chunk dari: " + ", ".join(hilang))
    if not idx.chunks:
        raise SystemExit("Indeks kosong: tidak ada teks yang terbaca dari folder dokumen/.")

    # Validasi soal: kunci jawaban harus benar-benar ada di korpus
    kunci = periksa_kunci(idx, PERTANYAAN + PARAFRASE)
```

```
soal_cacat = []
for s, n, file_ada in kunci:
    status = f"{n} chunk memuat kunci, di {', '.join(file_ada) or '-'}"
    if n == 0 or s["file"] not in file_ada:
        soal_cacat.append(s["q"])
        status += "  <-- [PERINGATAN] kunci tidak ada di file yang diharapkan"
    print(f"  soal: {s['q'][:60]}... | {status}")

# ---- Langkah 2
print("\n=== Langkah 2: 5 pertanyaan ===")
h2 = jalankan(idx, PERTANYAAN)
for s, atas, rel in h2:
    print(f"\nQ: {s['q']}  (kata kunci: {s['kw']})")
    for r, (a, ok) in enumerate(zip(atas, rel), 1):
        print(f"  #{r} skor={a['skor']:.3f} {sumber(a)} relevan={ok} | {ringkas(a['teks'])}")
n_top1 = sum(top1(rel) for _, _, rel in h2)
n_top3 = sum(any(rel) for _, _, rel in h2)
n_soal = len(PERTANYAAN)

# ---- Langkah 3
print("\n=== Langkah 3: parafrase ===")
sem = jalankan(idx, PARAFRASE)
lek = jalankan(idx_leksikal, PARAFRASE)
for (s, a1, r1), (_, a2, r2) in zip(sem, lek):
    print(f"\nQ: {s['q']}")
    print(f"  {idx.backend}: top1 relevan={top1(r1)} top3 relevan={any(r1)} | {sumber(a1[0] if a1
else None)}")
    print(f"  tfidf : top1 relevan={top1(r2)} top3 relevan={any(r2)} | {sumber(a2[0] if a2 else
None)}")
n_par = len(PARAFRASE)

# ---- Kesimpulan: setiap klaim dihitung dari data
sem_top3 = sum(any(r) for _, _, r in sem)
lek_top3 = sum(any(r) for _, _, r in lek)
sem_top1 = sum(top1(r) for _, _, r in sem)
lek_top1 = sum(top1(r) for _, _, r in lek)
kes = (f"Pada {n_soal} pertanyaan langsung, chunk peringkat 1 relevan pada {n_top1}/{n_soal}
pertanyaan dan "
       f"salah satu dari 3 chunk teratas relevan pada {n_top3}/{n_soal}. Pada {n_par} pertanyaan
parafrase, "
       f"backend '{idx.backend}' menemukan chunk relevan di peringkat 1 pada {sem_top1}/{n_par} dan
di 3 "
       f"teratas pada {sem_top3}/{n_par}, sedangkan pembanding TF-IDF {lek_top1}/{n_par} dan "
       f"{lek_top3}/{n_par}. ")

gagal = [(s, n) for (s, atas, rel), (_, n, _) in zip(h2, kunci[:n_soal]) if not top1(rel)]
for s, n in gagal:
    r = peringkat_relevan(idx, s)
    if n == 0:
        kes += (f'Untuk "{s["q"]}", tidak ada satu chunk pun yang memuat semua kata kunci '
                "(jawaban terbelah di batas chunk atau kata kunci keliru). ")
    elif r is None:
        kes += f'Untuk "{s["q"]}", chunk relevan tidak muncul sama sekali dalam peringkat indeks. '
    else:
        kes += (f'Untuk "{s["q"]}", chunk relevan pertama baru muncul di peringkat {r} dari '
                f"{len(idx.chunks)} chunk. ")
if not gagal:
    kes += "Semua pertanyaan langsung menemukan chunk relevan di peringkat 1. "

if sem_top1 == lek_top1 and sem_top3 == lek_top3:
    kes += ("Pada parafrase ini, hasil semantik dan TF-IDF sama, sehingga tes ini belum membedakan "
            "keduanya; gunakan parafrase yang tidak berbagi kata atau akar kata dengan teks untuk "
            "menguji kemampuan semantik. ")
elif (sem_top1, sem_top3) > (lek_top1, lek_top3):
    kes += "Pada parafrase ini, pencarian semantik lebih unggul daripada TF-IDF. "
else:
    kes += "Pada parafrase ini, TF-IDF justru sama baik atau lebih baik daripada pencarian semantik.
"
kes += (f"Korpus hanya {len(idx.chunks)} chunk dari {len(per_file)} file, sehingga satu pertanyaan "
       "mengubah persentase cukup besar. ")
if soal_cacat:
    kes += "Perhatian: kunci jawaban tidak ditemukan di file yang diharapkan untuk soal: " + \
           "; ".join(f'"{q}"' for q in soal_cacat) + ". "
if idx.backend == "lsa":
    kes += ("Catatan: backend LSA hanya semantik ringan (hubungan antar kata dipelajari dari korpus
yang "
            "kecil), sehingga parafrase yang tidak berbagi kata akan sulit. Untuk hasil lebih kuat,
pasang "
            "sentence-transformers; kode ini otomatis memakainya.")
kes = kes.strip()
print("\nKesimpulan:", kes)

# ---- Laporan PDF
```

```
    L = Laporan(os.path.join(BASE, "Lab2_Indeks_dan_Pencarian.pdf"), "Lab 2 - Bangun Indeks & Cari",
                f"ukuran chunk {UKURAN}, overlap {int(OVERLAP * 100)}%, backend: {idx.backend}")
    L.h1("Langkah 1. Membangun indeks")
    L.p(f"{len(per_file)} PDF berada di folder dokumen/. Setiap file melaporkan jumlah chunk-nya:")
    L.tabel([["File", "Jumlah chunk"]] + [[f, n] for f, n in per_file.items()] + [["Total",
    len(idx.chunks)]],
            [9 * 28.35, 4 * 28.35])
    if hilang:
        L.p("Peringatan: tidak ada chunk dari " + ", ".join(hilang) + ".")
    L.kode(log_indeks.rstrip())
    L.h1("Langkah 2. Lima pertanyaan dan pemeriksaan chunk teratas")
    L.p("Sebuah chunk dianggap relevan bila memuat semua kata kunci jawaban (huruf besar/kecil dan "
        "pemisah baris diabaikan). Tabel menunjukkan 3 chunk teratas.")
    for s, atas, rel in h2:
        L.h2(s["q"])
        L.tabel([["Peringkat", "Skor", "Sumber", "Relevan?", "Cuplikan"]] +
                [[r, f"{a['skor']:.3f}", sumber(a), ya(ok), ringkas(a["teks"], 140)]
                 for r, (a, ok) in enumerate(zip(atas, rel), 1)],
                [1.6 * 28.35, 1.3 * 28.35, 3.8 * 28.35, 1.6 * 28.35, 8.7 * 28.35])
    L.p(f"Ringkasan: chunk peringkat 1 relevan pada {n_top1}/{n_soal} pertanyaan; ada chunk relevan di 3
    teratas "
        f"pada {n_top3}/{n_soal}.")
    L.h1("Langkah 3. Pertanyaan berparafrase")
    L.p("Dua pertanyaan memakai kata yang berbeda dari teks dokumen. Dibandingkan: pencarian semantik "
        f"({idx.backend}) vs TF-IDF.")
    baris = [["Pertanyaan", "Catatan", f"{idx.backend}: top1 / top3", "TF-IDF: top1 / top3"]]
    for (s, a1, r1), (_, a2, r2) in zip(sem, lek):
        baris.append([s["q"], s["catatan"], f"{ya(top1(r1))} / {ya(any(r1))}", f"{ya(top1(r2))} /
    {ya(any(r2))}"])
    L.tabel(baris, [5 * 28.35, 6 * 28.35, 3 * 28.35, 3 * 28.35])
    for (s, a1, r1), _ in zip(sem, lek):
        if a1:
            L.p(f"Chunk teratas untuk \"{s['q']}\": {sumber(a1[0])} (skor {a1[0]['skor']:.3f})")
            L.kode(ringkas(a1[0]["teks"], 400))
    L.h1("Kesimpulan")
    L.p(kes)
    L.halaman_baru()
    L.h1(f"Lampiran: kode {os.path.basename(__file__)}")
    L.kode(open(__file__, encoding="utf-8").read())
    L.simpan()
    print("\nLaporan: Lab2_Indeks_dan_Pencarian.pdf")


if __name__ == "__main__":
    main()
```