Project Object Oriented Programming

Praktikum Pemrograman Dasar – Kelompok 9 Teknik Komputer D 2025

---

1. Identitas Anggota Kelompok 9

- JOSE IMANUEL STANLEY WIJAYA/255150300111040
- QUEEN NADIYA AILSA DIKLA/25515030111029
- MUHAMMAD HANIF FADHILAH/25515030111026

2. Latar Belakang

Program ini dibuat untuk mencatat identitas guru, murid, serta nilai tugas. Dalam kegiatan belajar, pencatatan manual sering menyebabkan data berantakan dan sulit dicari. Dengan memanfaatkan konsep *Object-Oriented Programming* seperti class, inheritance, dan array statis, program ini menyediakan sistem terstruktur untuk menyimpan data guru, murid, jumlah tugas, serta perhitungan rata-rata nilai. Pembatasan kapasitas melalui konstanta (MAX_STUDENTS, MAX_TASKS, MAX_TEACHERS) memastikan data terkendali dan mudah diolah. Program ini menjadi simulasi sederhana sistem penilaian di sekolah yang membuat input nilai lebih mudah, memantau perkembangan murid, dan menampilkan laporan akhir secara rapi.

3. Penjelasan Program

Penjelasan Deskripsi

MAX_STUDENTS = 10: jumlah slot murid tetap (index 0..9).
MAX_TASKS = 5: maksimal jumlah tugas per murid.
MAX_TEACHERS = 5: jumlah slot guru tetap (index 0..4).
Fungsi: kontrol kapasitas dan ukuran array statis.
Class Person

anggota protected:
string nama: menyimpan nama (guru atau murid).
int umur: menyimpan umur.
public:
Person(): konstruktor default (nama = "", umur = 0).
void setData(const string& n, int u): set nama & umur.
string getNama() const / int getUmur() const: getter untuk membaca data.
Fungsi: kelas dasar dipakai sebagai basis Teacher dan Student.

Class Student : public Person

private:
int tugas[MAX_TASKS]: array nilai tugas, indeks 0..MAX_TASKS-1.
int nTugas: berapa tugas yang sudah diisi.
public:
Student(): konstruktor, inisialisasi nTugas = 0 dan tugas di-set 0.
void init(const string& n, int u): set nama & umur (panggil Person::setData) dan reset nTugas.
int submit(int nilai): tambahkan nilai ke tugas jika belum penuh; return 1 sukses, 0 jika penuh.
double rata() const: hitung rata-rata nilai tugas; jika nTugas==0 kembalikan 0.0.
int jumlahTugas() const: kembalikan nTugas.
int nilaiAt(int i) const: kembalikan tugas pada indeks i (atau 0 bila indeks invalid).
Fungsi: menyimpan data murid dan nilai tugas, menghitung rata-rata.
Class Teacher : public Person

public:
Teacher(): konstruktor default.
void set(const string& n, int u): set nama & umur.
Fungsi: menyederhanakan penyimpanan data guru.
Bagian main() — inisialisasi

Teacher teachers[MAX_TEACHERS]; lalu langsung di-set 5 guru awal (nama & umur).
Ini artinya program sudah punya 5 slot guru yang bisa dipilih; tidak perlu input nama guru saat runtime (kecuali ada fitur edit).
Student siswa[MAX_STUDENTS]; array murid (semua kosong awalnya).
int currentTeacher = 0; index guru aktif (default slot 0).
Komentar singkat di main:

menjelaskan tujuan variabel teachers, siswa, currentTeacher.
Loop menu utama (while true)

Menampilkan menu pilihan 1..5 dan membaca int pilihan dengan cin >> pilihan.
Catatan: kode tidak mengecek input bukan-integer (bila user masukkan teks, cin akan gagal — program tidak menangani ini secara eksplisit).
Pilihan 1 — Lihat daftar guru

Loop i dari 0 sampai MAX_TEACHERS-1 dan tampilkan:
index i, teachers[i].getNama(), umur teachers[i].getUmur()
Fungsi: lihat nama/umur semua slot guru.
Pilihan 2 — Pilih guru aktif

Minta input nomor no; validasi: 0 <= no < MAX_TEACHERS.

Jika valid, set currentTeacher = no dan tampilkan nama guru aktif.
Fungsi: hanya mengubah indeks guru aktif untuk info tampil; data murid tetap global (satu daftar).
Pilihan 3 — Submit nilai (utama)

Minta nomor murid: int nomor; validasi 0..MAX_STUDENTS-1.
Cek apakah siswa[nomor] sudah pernah diisi: menggunakan siswa[nomor].getNama().size() == 0 untuk mendeteksi kosong.
Jika kosong: minta Nama (getline) dan Umur (cin), lalu panggil siswa[nomor].init(nama, umur).
Jika sudah ada: tampilkan "Murid terpilih: <nama>".
Input nilai tugas:
Loop meminta input nilai sampai valid.
Aturan validasi:
Jika nilai == 100 -> tampilkan "kesempurnaan hanya milik tuhan" lalu ulang input.
Jika nilai < 1 atau > 99 -> tampilkan "Nilai tidak valid. Masukkan nilai antara 1 sampai 99." lalu ulang.
Jika 1..99 -> terima dan keluar loop.
Setelah mendapatkan nilai valid, panggil siswa[nomor].submit(nilai):
Jika return 1 -> sukses simpan.
Jika return 0 -> slot tugas penuh (tidak disimpan).
Fungsi: menambahkan nilai tugas ke murid; aturan khusus menolak 100.
Pilihan 4 — Cetak nilai akhir (untuk guru aktif)

Tampilkan header dengan nama guru aktif (teachers[currentTeacher].getNama()).
Loop semua slot murid (i 0..MAX_STUDENTS-1):
Lewatkan jika nama murid kosong (getNama().size() == 0).
Untuk setiap murid terisi tampilkan:
index, nama, jumlah tugas (jumlahTugas()), daftar nilai (nilaiAt(t) untuk tiap tugas), dan rata-rata (rata()).
Jika tidak ada murid terisi sama sekali, tampilkan pesan "Belum ada murid yang terdaftar."
Fungsi: menampilkan ringkasan nilai semua murid; guru aktif hanya sebagai informasi, data murid tidak dipisah per guru.
Pilihan 5 — Keluar

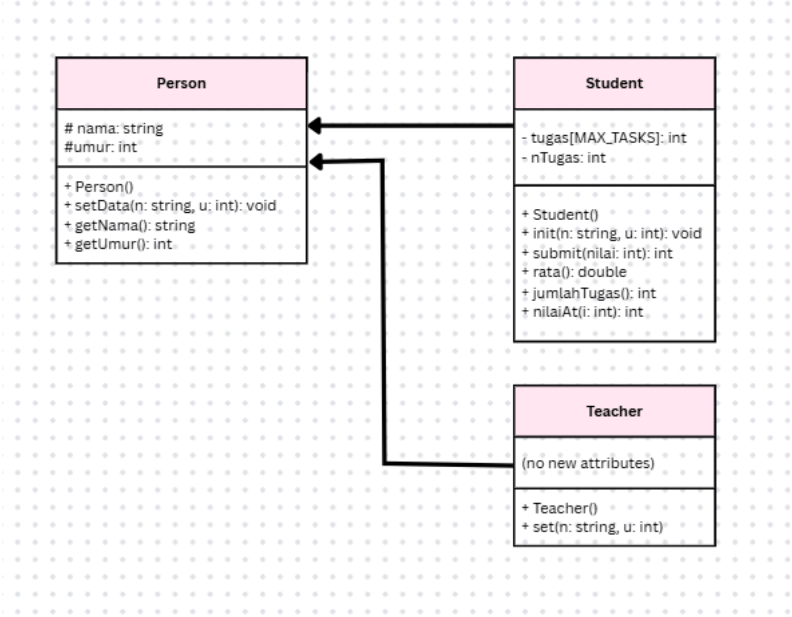
Break dari loop dan program selesai.
Variabel lokal penting di main (fungsi & maksud)

pilihan: menyimpan input menu.
no: nomor guru untuk memilih guru aktif pada pilihan 2.
nomor: slot murid yang dipilih pada pilihan 3.
nama, umur (lokal): input sementara untuk data murid baru.
nilai: nilai tugas yang diinput dan divalidasi.

hasil: return dari submit (1 sukses, 0 penuh).
jumlahDitemukan: hitung berapa murid yang terisi saat cetak.
Perilaku & catatan penting

Data murid bersifat global (satu daftar). Mengganti currentTeacher tidak mengubah nilai murid.
Setiap murid punya batas tugas MAX_TASKS; submit() mencegah overflow.
Rata-rata dihitung hanya dari tugas yang terisi; jika belum ada tugas, rata() mengembalikan 0.0.
Pemeriksaan "kosong" murid dilakukan lewat getNama().size() == 0.
Input validation minimal:
Program memeriksa range index dan range nilai, juga menolak nilai == 100 khusus.
Program tidak menangani kasus input non-numerik (mis. kalau user mengetik huruf saat program menunggu int, cin akan gagal). Ini bisa diperbaiki nanti jika mau.
Pesan khusus: bila user masukkan nilai 100, program menampilkan "kesempurnaan hanya milik tuhan" dan meminta ulang nilai (sesuai permintaan).

UML Class Diagram



Pseudocode

```
// PSEUDOCODE for Pendidikan.cpp

// constants
SET MAX_STUDENTS = 10
SET MAX_TASKS = 5
SET MAX_TEACHERS = 5
```

```
// class Person
CLASS Person
    FIELD name : string
    FIELD age : int
    METHOD setData(name, age)
    METHOD getName() RETURNS name
    METHOD getAge() RETURNS age

// class Student inherits Person
CLASS Student EXTENDS Person
    FIELD tasks[MAX_TASKS] : int
    FIELD taskCount : int
    CONSTRUCTOR: initialize taskCount = 0, tasks[] = 0
    METHOD init(name, age) -> call setData and reset taskCount
    METHOD submit(score) RETURNS int
        IF taskCount >= MAX_TASKS THEN RETURN 0
        tasks[taskCount] = score
        taskCount = taskCount + 1
        RETURN 1
    METHOD average() RETURNS double
        IF taskCount == 0 THEN RETURN 0.0
        SUM all tasks[0..taskCount-1] and RETURN sum / taskCount
    METHOD getTaskCount() RETURNS taskCount
    METHOD getTaskAt(i) RETURNS tasks[i] or 0 if invalid

// class Teacher inherits Person
CLASS Teacher EXTENDS Person
    METHOD set(name, age) -> call setData

// main
FUNCTION main()
    CREATE array teachers[MAX_TEACHERS] of Teacher
    // preset teacher data
    teachers[0].set("Pak Heru", 40)
    teachers[1].set("Pak Arief", 38)
    teachers[2].set("Ibu Anes", 35)
    teachers[3].set("Pak Ai", 42)
    teachers[4].set("Bu Hanif", 30)

    CREATE array students[MAX_STUDENTS] of Student
    SET currentTeacher = 0

    LOOP forever
        PRINT menu:
            1. View teachers
            2. Select active teacher
            3. Submit score (enter student data if empty)
```

```
        4. Print final scores (show active teacher)
        5. Exit
      READ choice

      IF choice == 1 THEN
        FOR i FROM 0 TO MAX_TEACHERS-1 DO
          PRINT i, teachers[i].getName(), teachers[i].getAge()
      ELSE IF choice == 2 THEN
        PRINT "Enter teacher index 0..MAX_TEACHERS-1"
        READ index
        IF index invalid THEN PRINT error ELSE currentTeacher = index; PRINT selected
teacher
      ELSE IF choice == 3 THEN
        PRINT "Enter student slot 0..MAX_STUDENTS-1"
        READ slot
        IF slot invalid THEN PRINT error; CONTINUE loop
        IF students[slot].getName() is empty THEN
          PROMPT for student name and age
          students[slot].init(name, age)
        ENDIF
        // input score with validation:
        LOOP
          PRINT "Enter score (1-99). If 100 -> special message and re-enter."
          READ score
          IF score == 100 THEN
            PRINT "kesempurnaan hanya milik tuhan"
            CONTINUE loop
          ENDIF
          IF score < 1 OR score > 99 THEN
            PRINT "Invalid score, enter 1..99"
            CONTINUE loop
          ENDIF
          BREAK loop
        ENDLOOP
        result = students[slot].submit(score)
        IF result == 1 THEN PRINT "Score saved"
        ELSE PRINT "Tasks full for this student"
      ELSE IF choice == 4 THEN
        PRINT header with teachers[currentTeacher].getName()
        SET found = 0
        FOR i FROM 0 TO MAX_STUDENTS-1 DO
          IF students[i].getName() is not empty THEN
            found = found + 1
            PRINT student index, name, task count
            PRINT list of task scores
            PRINT average = students[i].average()
          ENDIF
        ENDFOR
```

```
        IF found == 0 THEN PRINT "No students registered"
      ELSE IF choice == 5 THEN
        PRINT "Exit" and BREAK loop
      ELSE
        PRINT "Invalid choice"
      ENDIF
    ENDLOOP

END FUNCTION
```

4. Source Code

https://github.com/queennadiya/Project-OOP-Kelompok-9.git