Debugowanie programu

Czas: 2 × 45 min | praca indywidualna | Wykład 1.12

Cel: Znajdziesz i poprawisz sześć błędów logicznych w gotowym programie, korzystając z debugera Visual Studio zamiast zgadywania.

Po zajęciach potrafisz:

- ustawia punkty przerwania i wykonuje program krokowo (F10, F11)
- obserwuje wartości zmiennych w oknach Locals i Watch
- używa warunkowego punktu przerwania
- opisuje błąd: objaw, przyczynę i poprawkę

Sprzęt i materiały

- komputer z Windows 11 i dostępem do internetu
- pendrive lub konto w repozytorium do zapisania projektu

Oprogramowanie

- **Visual Studio Community 2026 (obciążenie „Programowanie aplikacji klasycznych platformy .NET")** — darmowe dla uczniów i nauczycieli; w pracowni zainstalowane visualstudio.microsoft.com/pl/downloads
- **.NET 10 SDK (LTS)** — instalowany razem z Visual Studio; sprawdzenie: `dotnet --version` dotnet.microsoft.com/pl-pl/download/dotnet/10.0

Przygotowanie przed zajęciami

- Pobierz plik `Program.cs` z folderu `pliki` przy tym zadaniu.

Bezpieczeństwo

- Zapisuj projekt w swoim folderze (np. `C:\Users\<login>\source\repos`), a na koniec zajęć skopiuj go na pendrive lub wyślij do repozytorium.
- Nie uruchamiaj programów i pakietów z nieznanych źródeł; pakiety NuGet instaluj tylko te, które wskazuje zadanie.

Przebieg zadania

1 Uruchomienie programu z błędami

Utwórz projekt konsolowy `Debugowanie` i zastąp `Program.cs` plikiem od nauczyciela.

1. Przeczytaj komentarz na początku pliku — opisuje, co program **powinien** robić.
2. Oblicz na kartce poprawne wyniki dla danych z programu (średnia, liczba pozytywnych, procent, najlepsza ocena, imię wspak).
3. Uruchom program (Ctrl+F5) i porównaj wyniki z obliczeniami. Zapisz objawy błędów w tabeli.

2 Narzędzia debugera

Przećwicz narzędzia na pierwszej pętli.

1. Ustaw punkt przerwania (F9) na wierszu `suma += oceny[i];` i uruchom program klawiszem F5.
2. Wykonuj program krokowo (F10). W oknie **Locals** obserwuj `i` i `suma` — od jakiej wartości zaczyna się `i`?
3. Dodaj do okna **Watch** wyrażenie `oceny[i]` oraz `suma / oceny.Length`.
4. Najedź kursorem na zmienną w kodzie, aby zobaczyć jej wartość (DataTip).

3 Szukanie błędów

Znajdź wszystkie 6 błędów. Każdy popraw dopiero po znalezieniu przyczyny w debugerze.

1. Dla każdego złego wyniku ustaw punkt przerwania w miejscu obliczenia i sprawdź wartości.
2. Ustaw warunkowy punkt przerwania (prawy przycisk na czerwonej kropce → Warunki) na `o == 2` w pętli liczącej oceny pozytywne.
3. W oknie **Immediate** (Debuguj → Okna → Natychmiastowe) wpisz `100.0 * pozytywne / oceny.Length` i porównaj z wynikiem programu.
4. Przy pętli odwracającej imię obserwuj ostatnią wartość `i` przed wyjściem z pętli.
5. Po każdej poprawce uruchom program i zapisz wynik w tabeli.

4 Zakończenie

Sprawdź, czy wszystkie wyniki są zgodne z obliczeniami na kartce.

1. Opisz każdy błąd w tabeli sprawozdania.
2. Zatwierdź poprawioną wersję w Git z opisem „Poprawione 6 błędów”.

W sprawozdaniu umieść

- ☐ tabela 6 błędów: objaw, przyczyna, poprawka
- ☐ zrzut ekranu debugera z oknem Locals lub Watch
- ☐ zrzut ekranu warunkowego punktu przerwania
- ☐ wyniki programu po poprawkach

Pytania kontrolne

1. Czym różni się F10 (Step Over) od F11 (Step Into)?
2. Do czego służy warunkowy punkt przerwania? Podaj przykład spoza zadania.
3. Dlaczego kompilator nie wykrył żadnego z tych błędów?
4. Jak test jednostkowy mógłby wykryć błąd w obliczaniu średniej?

Kryteria oceny

Kryterium	Punkty
Obliczenia na kartce i objawy błędów	3
Znalezione i poprawione błędy (po 2 pkt)	12
Zrzuty z debugera	2
Pytania	3
Razem	**20**

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).