

# Tworzenie oprogramowania

## – rys historyczny

- gwałtowny rozwój technik komputerowych
- "kryzys oprogramowania"
  - duża złożoność systemów
  - niepowtarzalność przedsięwzięć
  - nieprzejrzystość procesu tworzenia
  - pozorna łatwość tworzenia kodu
- inżynieria oprogramowania, czyli jak tworzyć i utrzymywać "dobre" programy

# **Zakres inżynierii oprogramowania**

- prowadzenie przedsięwzięć informatycznych
- metody analizy i projektowania
- przygotowywanie dokumentacji technicznej i użytkowej
- sposoby zwiększania niezawodności, metody testowania
- procedury kontroli jakości
- redukcja kosztów konserwacji
- techniki pracy zespołowej
- narzędzia CASE

etc.

# **Etapy tworzenia systemu komputerowego**

1. Analiza (potrzeby → model systemu)
2. Projektowanie (→ projekt systemu)
3. Programowanie (→ działający system)
4. Testowanie (→ poprawny system)
5. Wdrażanie

# **Możliwe cykle tworzenia oprogramowania**

- exploratory programming
- waterfall model
- document-driven realisation
- prototyping
- incremental development

# Potencjalne problemy

- komunikacja  
(wewnętrzna i zewnętrzna)
- różne metodologie i standardy
- zmienne wymagania
- istniejące ograniczenia

etc.

# Ewolucja technologii tworzenia systemów komputerowych

**Cel:** umożliwienie lub ułatwienie

- tworzenia dużych systemów
- wielokrotnego wykorzystania opracowanych rozwiązań
- wprowadzania modyfikacji
- tworzenia systemów rozproszonych
- tworzenia systemów czasu rzeczywistego
- pracy w dużych grupach ludzkich
- kontynuacji projektu przez nowe (inne) grupy pracowników

etc.

# Paradygmat obiektowy

– opanowywanie złożoności (dziedziny) problemu opiera się na powszechnie stosowanych zasadach:

- rozróżnianie poszczególnych obiektów
- tworzenie pojęć – łączenie obiektów w klasy
- znajdowanie zależności między pojęciami – klasami, wyprowadzanie nowych
- określanie relacji między obiektami, agregacja, kojarzenie

etc.