RODZAJE ŚRODOWISK PROGRAMISTYCZNYCH

Podstawy Programowania

Środowiska programistyczne

- Środowiska programistyczne, znane również jako Integrated Development Environments (IDEs), są narzędziami, które ułatwiają programistom pisanie, testowanie, debugowanie oraz wdrażanie oprogramowania. Istnieje wiele różnych rodzajów środowisk programistycznych, z których każde ma swoje cechy i zastosowania. Najbardziej nam znane to zdecydowanie:
  - ***Tekstowe Edytory Kodu***
  - ***Zintegrowane Środowiska Programistyczne***
- Choć możemy to rozwinąć na więcej kategorii niż wyłącznie na te podane.

Tekstowe Edytory Kodu

- Tekstowe edytory kodu są popularne ze względu na swoją prostotę i lekkość. Są często używane przez programistów, którzy preferują minimalistyczne narzędzia lub pracują nad mniejszymi projektami. Główne cechy to:
  - ***Rozszerzalność:***
    - Tekstowe edytory są często rozszerzane za pomocą wtyczek, co pozwala dostosować je do konkretnych potrzeb programisty.
  - ***Łatwość Użycia:***
    - Są zazwyczaj łatwe do nauki i szybkie w użyciu, co sprawia, że są popularne wśród początkujących programistów.
  - ***Szybkość i Wydajność:***
    - Dzięki minimalistycznemu interfejsowi działają szybko, nawet na starszych komputerach.
- *Przykładowe edytory tekstu:*
  - *Sublime Text, Visual Studio Code, Atom*

Zintegrowane Środowiska Programistyczne

- Zintegrowane Środowiska Programistyczne (IDEs) są kompleksowymi narzędziami, które oferują wiele funkcji ułatwiających pracę programistyczną. Są często używane w profesjonalnych projektach programistycznych. Główne cechy to:
  - ***Wsparcie dla Wielu Języków:***
    - IDE często oferują wsparcie dla różnych języków programowania, co pozwala programistom pracować z różnorodnymi technologiami.
  - ***Integracja Narzędzi:***
    - Zawierają wbudowane narzędzia, takie jak kompilatory, debuggery, analizatory statyczne i zarządzanie projektem.
  - ***Automatyzacja Zadań:***
    - Pozwalają na automatyzację wielu zadań, takich jak kompilacja, testowanie i wdrażanie, co zwiększa efektywność pracy.
- *Przykładowe IDE:*
  - *IntelliJ IDEA, Eclipse, PyCharm*

Środowiska do Nauki Programowania

- Środowiska te są przeznaczone dla początkujących programistów i dzieci, aby ułatwić naukę podstaw programowania. Główne cechy to:
  - ***Wizualne Programowanie:***
    - Wykorzystują bloki kodu lub interfejsy przypominające puzzle, które uczą podstawowych koncepcji programowania w intuicyjny sposób.
  - ***Edukacyjne Zastosowania:***
    - Są często używane w szkołach i na warsztatach edukacyjnych do nauki podstaw programowania bez konieczności nauki skomplikowanej składni języków programowania.
- *Przykładowe środowiska do nauki:*
  - *Scratch*, *Blockly*, *Thonny*

Środowiska Do Pracy Z Konkretnymi Językami

- Te środowiska są zoptymalizowane do pracy z konkretnymi językami programowania lub platformami. Główne cechy to:
  - ***Specyficzne Funkcje:***
    - Oferują funkcje specyficzne dla danego języka, które ułatwiają pracę z danym językiem programowania.
  - ***Integracja Z Narzędziami:***
    - Często integrują się z bibliotekami i narzędziami używanymi w danym języku, ułatwiając rozwój aplikacji.
- *Przykłady:*
  - *IDLE dla Pythona, Xcode dla Swifta, Android Studio dla Androida*

Środowiska Do Rozwoju Stron Internetowych

- Te środowiska są zaprojektowane specjalnie dla programistów pracujących nad projektami internetowymi. Główne cechy to:
  - ***Podświetlanie Składni:***
    - Podświetlanie składni HTML, CSS i JavaScript pomaga programistom w pisaniu poprawnego kodu.
  - ***Podgląd Na Żywo:***
    - Oferują funkcję podglądu na żywo, co pozwala programistom widzieć, jak zmiany w kodzie wpływają na wygląd strony.
- *Przykłady:*
  - *Visual Studio Code, WebStorm, Sublime Text z wtyczkami*

Środowiska Do Analizy Danych i Nauki Maszynowej

- Środowiska te są dedykowane dla naukowców danych i analityków, którzy pracują nad analizą danych, tworzeniem modeli statystycznych oraz uczeniem maszynowym. Główne cechy to:
  - ***Interaktywne Notatki:***
    - Pozwalają na tworzenie interaktywnych notatek, które zawierają kod, wyniki i wykresy, ułatwiając eksploracyjną analizę danych.
  - ***Wsparcie Dla Różnych Języków:***
    - Oferują wsparcie dla różnych języków programowania używanych w analizie danych, takich jak Python, R czy Julia.
- *Przykłady:*
  - *Jupyter Notebook*, *RStudio*, *Anaconda*

Środowiska Do Programowania Mikrokontrolerów

- Te środowiska są zaprojektowane specjalnie dla programistów pracujących nad projektami z wykorzystaniem mikrokontrolerów, które są małymi układami mikroprocesorowymi. Główne cechy środowisk do programowania mikrokontrolerów to:
  - *Symulatory Urządzeń:*
    - Oferują symulatory, które pozwalają programistom testować kod bez konieczności rzeczywistego podłączania mikrokontrolerów. To przyspiesza i ułatwia procesy testowania i debugowania.
  - *Wsparcie dla Wielu Platform:*
    - Obsługują różne rodzaje mikrokontrolerów, takie jak Arduino, Raspberry Pi, ESP8266 itp. Każda platforma może mieć swoje własne środowisko programistyczne lub wsparcie w istniejących IDEs.

Środowiska Do Programowania Mikrokontrolerów

  - *Wizualizacja Pinów i Interfejsów:*
    - W przypadku mikrokontrolerów ważne jest zrozumienie, które piny są używane do jakich funkcji. Środowiska te często oferują wizualizacje pinów, ułatwiając podłączanie różnych urządzeń do mikrokontrolera.
  - *Biblioteki i Przykłady:*
    - Wiele środowisk zawiera biblioteki i przykłady kodu, które pomagają programistom w szybkim rozpoczęciu pracy z różnymi urządzeniami i sensorami.
  - *Komunikacja Z Hardware'em:*
    - Pozwalają na łatwą komunikację z mikrokontrolerem, umożliwiając wgrywanie kodu na urządzenie oraz odbieranie danych zwrotnych.
- *Przykłady:*
  - *Arduino IDE*, *PlatformIO*, *MPLAB X IDE*