**Przedmiot: Podstawy informatyki (dr inż. Witold Alda)**

1. Jednym z prostych sposobów zapisu algorytmu jest schemat blokowy. Proszę podać zasady tworzenia schematu blokowego.

***Jednym z prostych i intuicyjnych sposobów przedstawienia algorytmu jest schemat blokowy.*** *Schemat blokowy to graficzna forma zapisu algorytmu, która przedstawia kolejne kroki w postaci połączonych symboli. Aby poprawnie stworzyć schemat blokowy, należy przestrzegać kilku podstawowych zasad:*Elipsa Start / Stop Początek i koniec algorytmu  
Prostokąt Działanie (proces) Wykonanie operacji, np. obliczenia  
Równoległobok Wejście / Wyjście Odczyt danych, wypisanie wyniku  
Romb Decyzja (warunek) Sprawdzenie warunku logicznego  
Strzałka Przepływ Określa kierunek przepływu sterowania

2. Proszę omówić znaczenie pojęć: składnia i semantyka języka programowania.

**Składnia (ang. syntax)**  
Składnia określa **zasady poprawnego zapisu kodu źródłowego** w danym języku programowania. Obejmuje m.in.:  
kolejność słów kluczowych, sposób definiowania instrukcji, zmiennych, funkcji, użycie nawiasów, przecinków, średników itp.Można powiedzieć, że składnia to „gramatyka” języka programowania. Jeśli kod ma błędy składniowe, nie zostanie skompilowany lub uruchomiony.

**Semantyka (ang. semantics)**  
Semantyka dotyczy **znaczenia instrukcji** – czyli tego, **co dany fragment kodu robi**.  
Nawet jeśli składnia jest poprawna, kod może robić coś innego niż zamierzał programista – i wtedy mamy do czynienia z błędem semantycznym.

3. Jednym z podstawowych pojęć w informatyce jest pojęcie zmiennej. Proszę podać cechy charakterystyczne zmiennej oraz omówić przynajmniej dwie klasy zmiennych.

Zmienne służą do przechowywania danych w pamięci komputera, które mogą być wykorzystywane i modyfikowane podczas działania programu.

Cechy charakterystyczne zmiennej:  
Nazwa – unikalny identyfikator, za pomocą którego odwołujemy się do zmiennej.  
Typ danych – określa, jaki rodzaj danych może być przechowywany (np. liczba całkowita, tekst, liczba zmiennoprzecinkowa).  
Wartość – aktualna zawartość zmiennej (może się zmieniać podczas działania programu).  
Zakres (ang. scope) – określa, gdzie w programie zmienna jest dostępna.  
Czas życia (ang. lifetime) – określa, jak długo zmienna istnieje w pamięci.

Lokalna, Globalna

4. Stos i sterta. Proszę omówić te pojęcia i na przykładzie w języku ANSI C pokazać kiedy i w jaki sposób korzystamy ze stosu, a kiedy ze sterty.

Stos i sterta to dwa różne obszary pamięci operacyjnej wykorzystywane podczas działania programu, w szczególności w języku C. Służą one do przechowywania danych, ale działają na innych zasadach i mają różne zastosowania.

Stos (ang. stack)  
Pamięć przydzielana automatycznie przez kompilator.  
Używany do przechowywania zmiennych lokalnych i danych funkcji (np. parametrów).  
Działa na zasadzie LIFO – ostatni przydzielony element jest pierwszy zwalniany.  
Szybki i wydajny, ale ma ograniczony rozmiar.

Sterta (ang. heap)  
Pamięć przydzielana dynamicznie przez programistę za pomocą funkcji malloc, calloc, realloc.  
Wymaga ręcznego zwolnienia pamięci (free()).  
Używana do alokacji danych, które muszą istnieć dłużej niż czas działania jednej funkcji (np. duże tablice, struktury).  
Ma większą pojemność niż stos, ale dostęp jest wolniejszy.

5. Omówić metody przekazywania argumentów do funkcji i pokazać je na przykładach w języku ANSI C.

W języku ANSI C istnieją dwie główne metody przekazywania argumentów do funkcji:  
Przekazywanie przez wartość (pass by value):  
Domyślna metoda w C. Do funkcji przekazywana jest kopią wartości zmiennej – oryginał nie zostaje zmieniony.  
Przekazywanie przez wskaźnik (symulacja pass by reference):  
Przekazujemy adres zmiennej, a nie jej wartość. Pozwala funkcji zmienić oryginalną zmienną. zmien(int \*x); zmien(&a);

**6. Rekurencja. Proszę omówić na czym polega i pokazać prosty przykład – można posłużyć się metajęzykiem.**

Rekurencja to technika programistyczna, w której funkcja wywołuje sama siebie, aby rozwiązać mniejsze instancje danego problemu. Jest to metoda dzielenia problemu na podproblemy tego samego typu, aż do osiągnięcia tzw. warunku brzegowego (czyli takiego, który nie wymaga dalszego wywoływania funkcji).

Warunek brzegowy (bazowy) – moment zatrzymania wywołań rekurencyjnych.  
Wywołanie rekurencyjne – funkcja wywołuje samą siebie z mniejszym lub zmienionym argumentem.

**Przedmiot: Metody probabilistyczne i statystyka**

**1. Miary tendencji centralnej i miary rozrzutu.**  
Podstawowe pojęcia w statystyce, które służą do opisu i analizy danych liczbowych.  
**Miary tendencji centralnej**  
Miary tendencji centralnej pokazują, wokół jakiej wartości koncentrują się dane – czyli jaka jest ich „środkowa” lub „typowa” wartość.  
Najczęściej stosowane miary:  
*Średnia arytmetyczna* – suma wszystkich wartości podzielona przez ich liczbę. *// (2+4+6)/3=4*  
*Mediana* – wartość środkowa po uporządkowaniu danych rosnąco. // 1,2, 3 mediana to 3,  
*Moda (dominanta)* – wartość występująca najczęściej. // 2,4,4,6, moda to 4.

**Miary rozrzutu (zmienności)**Miary rozrzutu pokazują, jak bardzo dane różnią się od siebie – czyli jak bardzo są „rozproszone” wokół wartości centralnej.  
Najczęściej stosowane miary:  
*Rozstęp* – różnica między największą a najmniejszą wartością.  
*Odchylenie standardowe* – informuje, jak bardzo wartości oddalają się od średniej.  
*Wariancja* – średnia z kwadratów odchyleń od średniej. Jest podstawą do obliczenia odchylenia standardowego.

**Średnia:** xˉ=(2+4+6+8+10)/5=6  
**Odchylenia od średniej:** (−4),(−2),0,2,4  
**Kwadraty odchyleń:** 16,4,0,4,16  
**Wariancja:** (16+4+0+4+16)/5=8  
**Odchylenie standardowe:** pierwiastek z 8≈2.83

**2. Omówić problem estymacji przedziałowej.**

**Estymacja przedziałowa** to metoda statystyczna, która pozwala oszacować nieznaną wartość w całej populacji (np. średnią) na podstawie próby. Zamiast podawać jedną liczbę, określa się **przedział ufności** – czyli zakres wartości, w którym z określonym **poziomem ufności** (np. 95%) znajduje się szacowany parametr.

Elementy estymacji przedziałowej  
Przedział ufności – przedział liczbowy, w którym z określonym prawdopodobieństwem znajduje się szacowany parametr.  
Poziom ufności (np. 90%, 95%, 99%) – prawdopodobieństwo, że przedział zawiera prawdziwą wartość parametru.  
**Estymacja przedziałowa** oznacza, że jeśli wiele razy przeprowadzilibyśmy podobne badanie, to w **95% przypadków** wyznaczone przedziały zawierałyby **prawdziwą wartość parametru** (np. średniej w populacji).

3. Scharakteryzować podstawowe pojęcia i omówić krótko procedurę weryfikacji hipotezy statystycznej.

Weryfikacja hipotezy statystycznej to jedna z podstawowych metod analizy danych, która pozwala sprawdzić, czy na podstawie wyników próby można potwierdzić przypuszczenia (hipotezy) dotyczące całej populacji.

Podstawowe pojęcia:  
Hipoteza zerowa (H0)  
– Jest to domyślna teza, którą testujemy i próbujemy obalić.

Hipoteza alternatywna (H1 lub Ha)  
– Twierdzenie przeciwne do hipotezy zerowej.

Poziom istotności (α)  
– Określa maksymalne dopuszczalne ryzyko popełnienia błędu I rodzaju.  
– Najczęściej: α = 0,05 (5%).

Wartość p (p-value)  
– Prawdopodobieństwo uzyskania takiego wyniku przy założeniu, że H0 jest prawdziwa.  
– Jeśli p < α, to odrzucamy H0.

Procedura  
H0: hipoteza zerowa  
H1: hipoteza alternatywna  
Wybór poziomu istotności α  
Dobór odpowiedniego testu statystycznego  
– np. test t-Studenta, test Z, test chi-kwadrat – zależnie od danych  
Obliczenie statystyki testowej i wartości p

Podjęcie decyzji:  
Jeśli p < α, odrzucamy Ha (hipoteza alternatywna jest prawdopodobna).  
Jeśli p ≥ α, brak podstaw do odrzucenia Ha.

**4. Statystyczne miary ilości informacji (entropia Shannona i miary pochodne).**W informatyce często chcemy wiedzieć, ile informacji zawiera jakiś komunikat. Do tego służą statystyczne miary informacji, a najważniejszą z nich jest entropia Shannona.  
Entropia Shannona  
Entropia to najważniejsza miara ilości informacji, wprowadzona przez Claude’a Shannona.  
Określa średnią ilość informacji, jaką niesie jedno zdarzenie lub symbol (np. znak w tekście).  
Co oznacza: Im bardziej losowe i nieprzewidywalne są dane, tym większa entropia.  
Im dane są powtarzalne i przewidywalne, tym entropia jest mniejsza.  
  
Miary pochodne od entropii  
Entropia warunkowa  
– Mierzy, ile niepewności zostaje, jeśli znamy inne dane.  
– Przykład: znajomość poprzedniego słowa może pomóc przewidzieć następne w zdaniu.

Wzajemna informacja  
– Mierzy, ile informacji jedno źródło przekazuje o drugim.  
– Przykład: ile wiedzy o X daje nam znajomość Y.

Dywergencja Kullbacka-Leiblera (KL)  
– Porównuje dwa rozkłady danych i mówi, jak bardzo się różnią.  
– Używana np. w uczeniu maszynowym.

Zastosowanie:  
Kompresja danych (np. ZIP, JPG), Szyfrowanie i bezpieczeństwo, Kodowanie sygnałów i transmisja danych

**5. Kompresja, kodowanie i szyfrowanie w informatyce.**

W informatyce pojęcia kompresji, kodowania i szyfrowania dotyczą przetwarzania i transmisji danych, ale każde z nich ma inny cel i zastosowanie.  
**Kompresja danych**  
Kompresja to proces zmniejszania rozmiaru danych w celu zaoszczędzenia miejsca na dysku lub przyspieszenia transmisji.  
Bezstratna – pozwala na dokładne odtworzenie oryginału. (ZIP, PNG, FLAC)  
Stratna – usuwa część danych, by zmniejszyć rozmiar kosztem jakości.(MP3, JPEG, MP4)  
zastosowanie: Przesyłanie plików przez Internet, Archiwizacja danych, Multimedia (dźwięk, obraz, wideo)  
**Kodowanie danych**  
Kodowanie polega na przekształcaniu danych do innej postaci – czytelnej dla maszyn lub ludzi, bez ukrywania treści.  
Przykłady: ASCII / Unicode – zamiana znaków na liczby binarne, Base64 – zamiana danych binarnych na ciąg znaków tekstowych (np. w e-mailach), Kodowanie Huffmana – używane w kompresji bezstratnej  
zastosowanie: Przechowywanie i transmisja danych, Formatowanie tekstów, plików, protokołów sieciowych  
**Szyfrowanie danych**  
Szyfrowanie to proces przekształcania danych w taki sposób, aby osoby nieuprawnione nie mogły ich odczytać. Stosuje się klucze (tajne lub publiczne) do zabezpieczania informacji.  
Symetryczne – ten sam klucz do szyfrowania i deszyfrowania (np. AES)  
Asymetryczne – różne klucze: publiczny i prywatny (np. RSA)  
zastosowanie: Bezpieczeństwo transmisji (np. HTTPS, VPN), Ochrona danych osobowych, Kryptografia

**Przedmiot: Języki formalne i kompilatory**

KOMPOLATOR

## ⚙️ 1. Wykonanie poleceń preprocesora  
- Obsługa dyrektyw preprocesora (np. `#include`, `#define`)  
- Usunięcie komentarzy  
- Wstawienie plików nagłówkowych  
## 🔎 2. Analiza leksykalna  
- Przekształca kod na tokeny (słowa kluczowe, identyfikatory, liczby)  
- Wykrywanie błędnych symboli  
## 🧩 3. Analizator syntaktyczny, Analiza składniowa (parsing)  
- Sprawdzenie zgodności z gramatyką języka  
- Tworzenie drzewa składniowego (AST)  
## 🧠 4. Analiza semantyczna  
- Sprawdzenie typów danych  
- Sprawdzenie, czy zmienne są zadeklarowane  
- Weryfikacja poprawności wyrażeń  
## 🚀 5. Optymalizacja kodu wynikowego  
- Usprawnianie kodu (np. eliminacja martwego kodu)  
- Redukcja liczby instrukcji  
## 🛠️ 6. Generowanie kodu  
- Tworzenie kodu maszynowego lub pośredniego (np. bytecode)  
- Przygotowanie do linkowania i wykonania

leksykalna->syntaktyczny->semantyczna

**1. Kompilacja i interpretacja, języki skryptowe i kompilowane, język Java.**

Kompilacja to proces, w którym cały kod źródłowy programu jest przetwarzany na kod maszynowy przed jego uruchomieniem. Kompilowane języki, takie jak C czy C++, wymagają wcześniejszego przetworzenia kodu źródłowego na plik wykonywalny. Interpretacja natomiast oznacza, że program jest przetwarzany i uruchamiany w trakcie jego działania przez interpreter, jak ma to miejsce w językach skryptowych, takich jak Python czy JavaScript. Java łączy cechy obu podejść: kod źródłowy jest najpierw kompilowany do kodu pośredniego (bytecode), który następnie jest interpretowany przez maszynę wirtualną JVM (Java Virtual Machine).

**2. Podstawowe moduły funkcjonalne kompilatora, rola kodu pośredniego przy kompilacji.**

Kompilator składa się z kilku podstawowych modułów funkcjonalnych: analizatora leksykalnego, analizatora syntaktycznego, analizatora semantycznego, generatora kodu pośredniego oraz optymalizatora i generatora kodu maszynowego. Kod pośredni, generowany po analizie syntaktycznej i semantycznej, stanowi abstrakcyjną formę programu, która może być optymalizowana i przekształcana na kod maszynowy odpowiedni dla różnych architektur sprzętowych.Użycie kodu pośredniego pozwala na większą przenośność i efektywność kompilacji.

**3. Wyjaśnić pojęcia: analiza leksykalna, wyrażenie regularne, automat skończony.**

Analiza leksykalna to proces, w którym kod źródłowy jest przetwarzany na mniejsze elementy zwane leksemami (tokeny). Analizator leksykalny grupuje ciągi znaków w logiczne jednostki, takie jak słowa kluczowe, identyfikatory, operatory czy literały. Wyrażenie regularne to formalny sposób opisywania wzorców w ciągach znaków i są one często wykorzystywane w analizie leksykalnej do identyfikowania leksemów. Automat skończony to model matematyczny służący do reprezentacji procesów o skończonej liczbie stanów, wykorzystywany w analizatorach leksykalnych do przetwarzania wejściowego ciągu znaków.

**4. Wyjaśnić pojęcia: analiza syntaktyczna, gramatyka, automat ze stosem.**Analiza syntaktyczna (parsing) to etap przetwarzania kodu źródłowego, w którym analizator syntaktyczny sprawdza, czy ciąg tokenów generowany przez analizator leksykalny spełnia zasady gramatyki języka programowania. Gramatyka to zbiór reguł definiujących poprawną strukturę wyrażeń w języku. Automat ze stosem to zaawansowany model automatu, który korzysta ze stosu do przechowywania dodatkowych informacji o stanie analizy, co pozwala mu na skuteczne przetwarzanie bardziej złożonych struktur językowych, takich jak zagnieżdżone nawiasy.

**5. Przykładowe techniki optymalizacji kodu pośredniego.**

Optymalizacja kodu pośredniego ma na celu poprawę efektywności kodu wynikowego poprzez redukcję nadmiarowych operacji lub wykorzystanie bardziej efektywnych instrukcji. Przykłady technik optymalizacji obejmują eliminację martwego kodu (usuwanie instrukcji, które nie wpływają na wynik), minimalizację liczby przejść do innych bloków kodu (redukowanie skoków) oraz zastępowanie złożonych wyrażeń prostszymi, które mają taką samą funkcjonalność (optymalizacja siłowa). Inne techniki to propagacja stałych, gdzie wartości stałe są wstawiane bezpośrednio do kodu zamiast zmiennych, oraz unikanie powtarzających się obliczeń.

- Metody probabilistyczne i statystyka (dr inż. Janusz Majewski; dr Beata Basiura)  
- Języki formalne i kompilatory (dr inż. Janusz Majewski; dr Edward Szczypka)  
- Bazy danych (dr inż. Robert Marcjan; dr inż. Marek Valenta)  
- Sztuczna inteligencja i systemy ekspertowe (dr inż. Marek Valenta; dr hab. inż. Rafał Dreżewski) **Framestick**  
- Zaawansowane systemy baz danych (dr inż. Marek Valenta ; dr inż. Robert Marcjan)  
- Architektury baz danych (mgr Grzegorz Stolecki)  
- Administracja i utrzymanie systemów baz danych (mgr Grzegorz Stolecki)  
- Grupowa sztuczna inteligencja (prof. dr hab. inż. Grzegorz Dobrowolski, dr inż. Adam Łuszpaj)  
JAVA Agent DEvelopment Framework (**JADE**)  
- Analiza i wizualizacja danych, Zastosowanie metod uczenia maszynowego Modelowanie danych wielowymiarowych   
 (mgr Grzegorz Stolecki)