**Algorytmy zagadnienia**

1. **Na przykładzie wektora 5 – 1 – 7 – 4– 3 wyjaśnij działanie algorytmu sortowania przez proste wstawianie. Kiedy lepszy jest algorytm sortowania przez proste wybieranie a kiedy przez proste wstawianie?**

* Bierzemy dowolny element z nieposortowanego wektora.
* Wyciągnięty element porównujemy z kolejnymi elementami zbioru posortowanego, póki nie napotkamy elementu równego lub elementu większego (jeśli chcemy otrzymać ciąg niemalejący) lub nie znajdziemy się na początku/końcu zbioru uporządkowanego.
* Wyciągnięty element wstawiamy w miejsce, gdzie skończyliśmy porównywać.
* Jeśli zbiór elementów nieuporządkowanych jest niepusty, zaczynamy kroki od początku wybierając inny element.

[5,1,7,4,3]🡪[1,5,7,4,3]🡪[1,4,5,7,3]🡪[1,3,4,5,7]

Nie należy do najszybszych ale :

* jest stabilny
* bardzo dobrze zachowuje się w przypadku zbioru posortowanego lub częściowo posortowanego
* prosty w implementacji
* dobrze radzi sobie z niedużymi zbiorami

Link:

<https://pl.wikipedia.org/wiki/Sortowanie_przez_wstawianie>

1. **Na przykładzie wektora 5 – 1 – 7 – 4– 3 wyjaśnij działanie algorytmu sortowania przez proste wybieranie. Kiedy lepszy jest algorytm sortowania przez proste wybieranie a kiedy przez proste wstawianie?**

* Wyszukujemy minimalną wartość z tablicy spośród elementów od i do końca tablicy.
* Zamieniamy wartość minimalną, z elementem na pozycji i.

[5,1,7,4,3]🡪[1,5,7,4,3]🡪[1,3,5,7,4]🡪[1,3,4,5,7]🡪[1,3,4,5,7]🡪

[1,3,4,5,7]

Zalety:

* wydajny dla danych wstępne posortowanych
* wydajny dla zbiorów o niewielkiej liczebności
* małe zasoby zajmowane podczas pracy (sortowanie w miejscu[1)](http://home.agh.edu.pl/~pkleczek/dokuwiki/doku.php?id=dydaktyka:aisd:2016:sort#fn__1))
* stabilność
* prostota implementacji

Wady:

* mała efektywność dla normalnej i dużej ilości danych

Link:

<https://pl.wikipedia.org/wiki/Sortowanie_przez_wybieranie>

1. **Na przykładzie wektora 5 – 1 – 7 – 4– 3 wyjaśnij działanie algorytmu sortowania bąbelkowego. Co to jest stabilność sortowania?**

* Porównujemy w kolejne elementy.
* Jeśli element n+1 jest większy od n, zamieniamy miejscami i przechodzimy do

kolejnych 2 elementów.

* Gdy dojdziemy do końca wektora, wracamy na sam początek.
* Czynność powtarzamy dopóki wszystkie elementy nie będą posortowane.

[5,1,7,4,3] 🡪[1,5,7,4,3]🡪[1,5,7,4,3]🡪[1,5,4,7,3]🡪[1,5,4,3,7]🡪[1,5,4,3,7]🡪

[1,4,5,3,7]🡪[1,4,3,5,7]🡪[1,4,3,5,7]🡪[1,4,3,5,7]🡪[1,3,4,5,7]🡪

[1,3,4,5,7]🡪[1,3,4,5,7]🡪[1,3,4,5,7]

**Stabilność**: **stabilne** algorytmy **sortowania** utrzymują kolejność występowania dla elementów o tym samym kluczu (klucz – cecha charakterystyczna dla każdego elementu zbioru, względem której jest dokonywane **sortowanie**).

Link:

<https://pl.wikipedia.org/wiki/Sortowanie_b%C4%85belkowe>

<https://www.szkolnictwo.pl/szukaj,Sortowanie>

**4. Wyjaśnij słowami zasadę działania algorytmu szybkiego sortowania**

Algorytm szybkiego sortowania należy do jednego z najszybszych algorytmów sortujących dane. Jest on chętnie implementowany i wdrażany do systemów informatycznych ze względu na szerokie spektrum.

**Działanie algorytmu szybkiego sortowania:**

Algorytm ten wykorzystuje technikę "dziel i zwyciężaj". Według ustalonego schematu wybierany jest jeden element w sortowanej tablicy, który będziemy nazywać **pivot.** Pivot może być elementem środkowym, pierwszym, ostatnim, losowym lub wybranym według jakiegoś innego schematu dostosowanego do zbioru danych. Następnie ustawiamy elementy nie większe na lewo tej wartości, natomiast nie mniejsze na prawo. W ten sposób powstaną nam dwie części tablicy (niekoniecznie równe), gdzie w pierwszej części znajdują się elementy nie większe od drugiej. Następnie każdą z tych podtablic sortujemy osobno według tego samego schematu.

* 1. **Algorytm a jego implementacja.**

Komputery przetwarzają przekazywane im informacje z wykorzystaniem algorytmów. Program jest algorytmem zapisanym w języku zrozumiałym dla maszyny (kodzie maszynowym). Każdy poprawny kod maszynowy da się przełożyć na zestaw instrukcji dla teoretycznego modelu komputera – maszyny Turinga.

Zwykle algorytmy pracują na danych wejściowych i uzyskują z nich dane wyjściowe. Informacje zapisane w pamięci maszyny traktuje się jako jej stan wewnętrzny. Niektóre algorytmy mają za zadanie wyłącznie przeprowadzanie komputera z jednego stanu wewnętrznego do innego.

Każdy algorytm komputerowy musi być wprowadzony do komputera w bardzo rygorystycznie zdefiniowanym języku. Ludzie często komunikując się, przesyłają między sobą informacje wieloznaczne. Komputery mogą reagować tylko na całkowicie jednoznaczne instrukcje. Jeżeli dany algorytm da się wykonać na maszynie o dostępnej mocy obliczeniowej i pamięci oraz w akceptowalnym czasie, to mówi się, że jest obliczalny.

Poprawne działanie większości algorytmów implementowanych w komputerach opiera się na kolejnej realizacji pewnego zestawu warunków. Jeżeli któryś z nich nie zostanie spełniony, to program kończy się komunikatem o błędzie. Czasami podczas implementacji algorytmu nawet istotny warunek może zostać pominięty. Przykładowo, w programie dzielącym przez siebie dwie liczby użytkownik poleca wykonać dzielenie przez zero. Działanie aplikacji, która nie sprawdzi warunku „czy dzielnik nierówny zero”, zostanie przerwane przez system operacyjny komputera.

Link:

<https://pl.wikipedia.org/wiki/Algorytm#Implementacja_algorytm%C3%B3w>

**6. Wymień i krótko scharakteryzuj podstawowe i dynamiczne struktury danych.**

**Struktury podstawowe** tak naprawdę nie zmieniają obszaru pamięci w trakcie realizacji programu.

**Podstawowe struktury danych**:

* Tablica
* Zbiór
* Rekord
* Plik sekwencyjny

**Dynamiczne struktury danych** – jest to metoda przechowywania danych przez program. Jest to bardzo ważne pojęcie z dziedziny programowania. Struktury dynamiczne polegają na tym, że dane przechowywane są w rezerwowanych na bieżąco miejscach pamięci powiązanych ze sobą przy użyciu wskaźników.

**Struktury dynamiczne:**

* Stos
* Kolejka
* Lista jednokierunkowa
* Lista dwukierunkowa
* Drzewo
* Sterty

**7. Podaj definicję tablicy. W jaki sposób tablice są tworzone, jak uzyskuje się dostęp do ich elementów?**

**Tablica** – jest to struktura danych przechowująca sekwencję wartości tego samego typu, która obsługuje tworzenie, dostęp indeksowy, przypisanie indeksowe oraz iteracje.

**W jaki sposób są tworzone?**

Tablice są tworzone w **3 etapach**. Są to min.:

* **Zadeklarowanie tablicy** – określamy jej nazwę oraz typ danych, jaki będzie przechowywać.
* **Utworzenie tablicy** – określamy jej wielkość (liczbę elementów).
* **Zainicjalizowanie elementów tablicy** – przypisujemy wartość do każdego elementu.

**Przykład tworzenia tablicy (język Java):**

String[] daysOfWeek = new String[7];

daysOfWeek[0] = "poniedziałek";

daysOfWeek[1] = "wtorek";

daysOfWeek[2] = "środa";

daysOfWeek[3] = "czwartek";

daysOfWeek[4] = "piątek";

daysOfWeek[5] = "sobota";

daysOfWeek[6] = "niedziela";

**Opis:**

Powyższy przykład pokazuje jak możemy stworzyć tablicę dla kilku elementów w języku Java. Pierwszym etapem jest tak naprawdę stworzenie naszej tablicy (w przykładzie posiada ona nazwę **daysOfWeek**) obiektów typy String (będziemy mogli w niej przechowywać tylko łańcuchy tekstowe). Następnie podaliśmy maksymalną ilość elementów, które nasza tablica będzie przechowywać (maksymalna ilość wynosi 7). Następnie dodajemy elementy do naszej tablicy, są one numerowane od 0 do 6 w tym przypadku.

**Jak uzyskuje się dostęp do ich elementów?**

Tak naprawdę dostęp do elementu w danej tablicy jest realizowany poprzez podanie jego pozycji (indeksów) w nawiasach kwadratowych.

**Przykład:**

np. tab[5], t[2][3]

**Linki:**

* <https://www.samouczekprogramisty.pl/tablice-w-jezyku-java/>

**8. Co to jest plik? Jakie są elementarne operacje na plikach?**

**Plik** – jest to tak naprawdę uporządkowany zbiór danych o skończonej długości, który posiada szereg atrybutów oraz stanowi dla użytkownika systemu operacyjnego jednolitą całość. Pliki możemy zapisywać na komputerze jak i na płytach czy na dyskach pen- drivie. Nie mamy tak naprawdę ograniczeń jeżeli chodzi o wielkość plików oraz ich zawartości. Muszą być one przechowywane w odpowiedni sposób, tak by nikt nieodpowiedni nie miał dostępu do zapisanych przez nas informacji. Obecnie pliki można przesyłać sobie za pomocą Internetu dzięki czemu trafiają do właściwego odbiorcy. Wszystkie pliki należy zachować gdyż mogą okazać się, że będziemy chcieli jeszcze z nich skorzystać i nie będziemy musieli tworzyć nowych.

Jakie są typy plików?

* **Katalogi** – są to tak naprawdę pliki zawierające spis odwołań do innych plików.
* **Pliki wykonywalne** – zawierają one głównie program, który musi być wykonany lub polecenia dla interpretera.
* **Kolejki FIFO** – są to pliki, które realizują bardziej skomplikowane zadania.
* **Dowiązania symboliczne** – inaczej mówiąc odwołanie do innego pliku.

Możemy tak naprawdę wyróżnić jeszcze kilka typów plików w **systemie DOS**:

* **Pliki tekstowe** – są to dane zapisane w formie kodowania ASCII łącznie z kodami sterującymi urządzeniami.
* **Pliki binarne** – są to inaczej pozostałe pliki.

**Jakie są elementarne operacje na plikach?**

* Tworzenie
* Zapis
* Odczyt
* Pozycjonowanie (seek) – inaczej przesuwanie kursora
* Usuwanie pliku
* Zerowanie pliku (zapis od zera)

**Dodatkowe operacje:**

* Dopisywanie na końcu pliku
* Zmiana nazwy pliku

**Linki:**

* <http://www.zpcir.ict.pwr.wroc.pl/~witold/opsys/os_fsys_h.pdf>
* <https://pl.wikipedia.org/wiki/Plik_danych>

**9. Struktury plików; sposoby wyróżniania pól.**

**Struktura** lub **rekord** — to złożony typ danych występujący w wielu językach programowania, grupujący logicznie powiązane ze sobą dane różnego typu w jednym obszarze pamięci. Składowe struktury — **pola** — są etykietowane, tj. mają swoje unikatowe nazwy; poprzez podanie nazwy otrzymuje się dostęp do danego pola.

Struktury są powszechnie stosowane w programowaniu, pozwalają bowiem w przejrzysty sposób opisywać złożone obiekty. Przykładem struktury może być informacja o książce, której pola będą zawierały: imię i nazwisko autora (typ łańcuchowy), tytuł (typ łańcuchowy), rok wydania (liczba całkowita), liczba stron (liczba całkowita), nazwa wydawnictwa, numer ISBN itp.

W językach z silną typizacją, zanim zostanie stworzona zmienna typu rekordowego, musi zostać zdefiniowany odpowiedni typ opisujący strukturę rekordu; takie typy nazywa się typami rekordowymi lub po prostu rekordami.

Podobną strukturą danych są krotki, w których pola nie są identyfikowane nazwą, lecz indeksem. W odróżnieniu jednak od krotek, dopuszcza się modyfikację pól struktury.

**10. Co to jest rekord? Porównaj rekord z tablicą.**

**Rekord** – jest to złożony typ danych występujący w wielu językach programowania, grupujący logicznie powiązane ze sobą dane różnego typu w jednym obszarze pamięci. Składowe struktury czyli pola, są etykietowane, tj. mają swoje unikatowe nazwy, poprzez podanie nazwy otrzymuje się dostęp do danego pola.

**Porównanie rekordu z tablicą:**

Rekord jest to pojedyncza jednostka przechowująca dane. Rekord może się zawierać w tablicy. Tablica potrafi przechowywać wiele rekordów jednocześnie.

**11. Listowe i tablicowe implementacje stosów, zalety i wady.**

**Listowa implementacja stosów:**

**Wady i zalety tablicowej implementacji stosów:**

* **Wady**
* Sposób jest tak naprawdę szybki oraz efektywny, ale posiada negatywną cechę, opiera się głównie na tablicy o stałej długości, w ten sposób ogranicza maksymalną wielkość stosu.
* Możliwość przepełnienia.
* **Zalety**
  + Łatwa implementacja.

**Wady i zalety listowej implementacji stosów:**

* **Wady**
* Trudniejsza implementacja w porównaniu do stosu opartego o tablicę.
* **Zalety**
  + Szybkość działania.
  + Możliwość przechowywania praktycznie nie ograniczonej ilości danych.

**12.** **Co to jest lista? Jakie są elementarne operacje zdefiniowane dla list?**

**Lista** – jest to struktura danych, która służy do reprezentacji zbiorów dynamicznych, w której elementy są ułożone w liniowym porządku.

**13. Wyjaśnij pojęcie stosu, kolejki i pierścienia. Co to są kolejki priorytetowe? Typowe implementacje stosów i kolejek. Problem Józefa.**

**Stos** – jest to tak naprawdę struktura liniowo uporządkowanych danych, z których jedynie ostatni element, zwany wierzchołkiem, jest w danym momencie dostępny. Głównie w wierzchołku odbywa się dołączanie nowych elementów, również jedynie wierzchołek można usunąć. Stos jest tak naprawdę często wykorzystywaną strukturą danych. Działanie na nim jest zazwyczaj porównywane do stosu talerzy: nie można usunąć talerza znajdującego się na dnie stosu nie usuwając wcześniej wszystkich innych. Nie można także dodać nowego talerza w inne miejsce, niż na samą górę.

**Kolejka** – jest ona pokrewna stosowi, lecz jej elementy są dodawane i usuwane z obu stron zgodnie z regułą **first-in, first-out (FIFO)**. Kolejkę opuszcza element, który jest w niej najdłużej.

**Pierścień** – jest to lista, w której ostatni element wskazuje na pierwszy.

**Co to są kolejki priorytetowe**:

**Kolejka priorytetowa** – jest to abstrakcyjny typ danych, który służy do reprezentowania zbioru elementów, z których każdy ma przyporządkowaną wartość zwaną kluczem.

**Zastosowania kolejek priorytetowych:**

Są stosowane w systemach komputerowych, przykładowo w systemach wielozadaniowych programy o najwyższych priorytetach powinny być uruchamiane jako pierwsze.

**Problem Józefa**:

**Problem Józefa Flawiusza** – jest to problem teoretyczny z zakresu kombinatoryki, jest często rozważany w informatyce. W ogólnej wersji problem brzmi następująco: na okręgu ustawiamy n obiektów, następnie eliminujemy co k-ty obiekt, tak długo, aż zostanie tylko jeden. Należy wskazać obiekt, który pozostanie. Dla k = 2 istnieje wzór jawny, a dla pozostałych k istnieją algorytmy rozwiązujące problem między innymi w złożonościach czasowych O(n) i O(k log n).

**Historia:**

Jakby mówiąc inaczej nazwa powyższego problemu nawiązuje do postaci Józefa Flawiusza, który w trakcie wojny rzymsko-żydowskiej został, wraz z grupą 40 żydowskich powstańców otoczony przez Rzymian w jaskini. Powstańcy od pojmania tak naprawdę woleli samobójstwo, dlatego też zdecydowali się utworzyć krąg i zabijać co trzecią osobę, aż zostanie tylko jedna, która popełni samobójstwo. Sam Józef Flawiusz wyliczył, w którym miejscu miał stanąć, aby uniknąć śmierci.

**Linki:**

* <https://eduinf.waw.pl/inf/alg/001_search/0100a.php>
* <https://pl.wikipedia.org/wiki/Stos_(informatyka)>

**14. Wyjaśnij pojęcie drzewa. Co to są drzewa zrównoważone i binarne.**

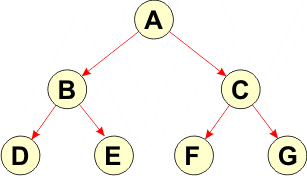
**Drzewo** – jest to struktura danych, która reprezentuje drzewo matematyczne. W inny sposób reprezentuje ono hierarchię danych (obiektów fizycznych i abstrakcyjnych). Drzewa ułatwiają i przyśpieszają wyszukiwanie, ale także pozwalają w łatwy sposób operować na posortowanych danych.

**Drzewo zrównoważone:**

**Drzewo zrównoważone** – jest to drzewo, w którym głębokość dowolnych dwóch liści różni się co najwyżej o jeden.

**Drzewo binarne**:

**Drzewo binarne** – jest to drzewo, w którym węzły mogą posiadać co najwyżej dwóch synów, nazywa się **drzewem binarnym**. Węzły potomne nazywamy odpowiednio **synem lewym** (left child node) i **synem prawym** (right child node). Drzewa binarne mają ogromne znaczenie w informatyce, ponieważ za ich pomocą można odwzorować również drzewa, których węzły posiadają dowolną liczbę synów.

A – korzeń, ojciec B i C  
B,C – synowie A, B – lewy syn A, C – prawy syn A  
D,E – liście, synowie B

F,G – liście, synowie C

**15. Na czym polega szukanie binarne?**

**Szukanie binarne:**

Szukanie binarne jest dość wydajnym algorytmem znajdowania elementu z uporządkowanej listy. Działa on poprzez wielokrotne dzielenie na pół fragmentu listy, który może zawierać szukany element, dopóki nie ograniczy jej tylko do jednej pozycji.

**Dokładny opis działania:**

Uporządkowana tablica jest dzielona na mniejsze przedziały do momentu, gdy przedział osiągnie długość jeden, kiedy jednym sprawdzeniem można określić, czy element znajduje się w tablicy.

W pojedynczym kroku rozważa się jeden przedział charakteryzowany dwoma indeksami: początkowym i końcowym. Algorytm zaczyna wyszukiwanie od całej tablicy.

Następnie jest wyznaczany środek przedziału:

**a** – indeks początkowy

**b** – indeks końcowy

Przedział jest zawężany – dzięki uporządkowaniu danych wiadomo, że albo poszukiwany element może znajdować się w indeksach [a, c] albo za nim. Inaczej mówiąc wybór ogranicza się do przedziału [a, c], gdy poszukiwany jest element mniejszy lub równy zapisanemu pod indeksem c lub [c + 1, b].

**16. Otwieranie i zamykanie plików, buforowanie.**

Co to jest buforowanie?

**Buforowanie** – jest to technologia polegająca na stosowaniu buforów programowych lub sprzętowych, jest ona najczęściej stosowana w celu wyrównania różnic prędkości przesyłania danych pomiędzy różnymi urządzeniami. Działa on zazwyczaj wykorzystując zasadę FIFO.

**Możemy wykorzystać bufory o**:

* Zerowej pojemności (bufor jest tylko przekaźnikiem komunikatorów)
* Ograniczonej pojemności
* Nieograniczonej pojemności

**17. Co to jest złożoność obliczeniowa? Jakie są klasy złożoności algorytmów? Przykłady notacji dużego ‘O’**

**Złożoność obliczeniowa** – jest to dział teorii obliczeń, którego głównym celem jest określanie ilości zasobów potrzebnych do rozwiązania problemów obliczeniowych.

**Jakie są klasy złożoności algorytmów?**

* **Klasa problemów (P)** – są to głównie problemy łatwe, dla których istnieje rozwiązać w czasie co najwyżej wielomianowym.
* **trudne (NP)** – są to problemy, dla których nie znamy rozwiązań w czasie wielomianowym lub mniejszym.
* podklasa problemów trudnych **NP-zupełne (NPC)** - czyli problemy zupełne w klasie NP ze względu na redukcje wielomianowe, to problemy, które należą do klasy NP oraz dowolny problem należący do NP może być do nich zredukowany w czasie wielomianowym. Jeśliby się udało rozwiązać problem NP-zupełny w czasie wielomianowym, to potrafilibyśmy rozwiązać w czasie wielomianowym wszystkie problemy NP. Problemy NP-zupełne można więc traktować jako najtrudniejsze problemy klasy NP (z punktu widzenia wielomianowej rozwiązywalności).

**Notacja dużego O** – notacja przedstawiająca **asymptotyczne tempo wzrostu**, wykorzystywana do zapisywania złożoności obliczeniowej algorytmu. Za pomocą tej notacji zapisywany jest rząd wielkości funkcji wyrażającej liczbę operacji dominujących (w przypadku złożoności czasowej) lub rozmiar wymaganej pamięci (w przypadku złożoności pamięciowej) w zależności od liczby danych wejściowych.

**Przykłady**:

* f(n) = n2 + 100n + log10 n + 1000 możemy przybliżyć jako: f(n) ≈ n2 + 100n + O(log10 n) albo jako: f(n) ≈ O(n2).
* ½ n2 - 3n = O(n2), ale również np. 5n +6 = O(n2).

**Linki:**

* <https://www.samouczekprogramisty.pl/podstawy-zlozonosci-obliczeniowej/>
* <http://strefakodera.pl/algorytmy/inne/zlozonosc-obliczeniowa-co-to-takiego>

**18. Kryteria oceny programów. Etapy programowania. Na czym polega uruchamianie a na czym testowanie?**

**Kryteria oceny programów:**

**Etapy programowania:**

* **Planowanie** – chodzi tutaj o sformułowanie problemu do rozwiązania, w tym etapie określamy cele projektu i zadania, które nasz program będzie realizować.
* **Tworzenie programu** – jest to etap, w którym przystępujemy do pisania programu, tworzymy kod źródłowy..
* **Kompilacja** – ten etap pozwala nam na sprawdzenie poprawności semantycznej i syntaktycznej programu oraz ustali rodzaj i lokalizację błędów.
* **Konsolidacja** – w procesie konsolidacji moduły programu, które powstały podczas kompilacji, są łączone w jeden program.
* **Testowanie** – testowanie programu to działanie zapewniające jakość oprogramowania. Ten etap jest stosowany w celu weryfikacji oraz walidacji oprogramowania.
* **Optymalizacja** – jest to działanie, którego celem jest poprawienie wydajności programu.

**Na czym polega uruchamianie, a na czym testowanie?**

**Uruchamianie** polega głównie na wykonaniu programu.

**Testowanie** polega na uruchamianiu programu lub aplikacji z zamiarem znalezienia błędów oprogramowania. Inaczej mówiąc polega ono na walidacji oraz weryfikacji, czy program działa zgodnie z zamierzeniem, czy może zostać uruchomiony mając niezmienioną charakterystykę.

**19. Abstrakcyjne struktury i typy danych i ich implementacja**

Co to jest abstrakcyjna struktura danych?

**Abstrakcyjna struktura danych** – jest to zbiór danych elementarnych wraz z dobrze zdefiniowanym na nich zbiorem operacji.

**20. Obiekty, klasy i metody. Klasy opakowujące. Konstruktory**.

**Obiekt** – jest to reprezentacja wartości określonego typu w pamięci komputera, charakteryzowana przez jego stan (wartość typu danych), zachowanie (operacje na typie danych) oraz tożsamość (lokalizację w pamięci).

**Metoda** – jest to nazwana sekwencja instrukcji, która może być wywołana przez inny kod w celu wykonania obliczeń.

**Klasa** – jest to konstrukcja języka, która służy do implementacji zdefiniowanego przez użytkownika typu danych, stanowiącego szablon do tworzenia obiektów zawierających wartości tego typu i manipulowania nimi.

**Klasy opakowujące**:

**Klasy osłonowe (opakowujące)** - są one jakby pomostem pomiędzy typami prostymi a obiektowymi. Ponieważ na typach prostych nie można wywoływać żadnych metod, jak robimy to na obiektach, wprowadzono klasy opakowujące, rozszerzające ich standardowe możliwości. Przykładowo obiekt typu Double reprezentujący liczby zmiennoprzecinkowe, można łatwo przekonwertować do typu long lub zamienić na string.

**Klasy osłonowe (opakowujące)** powstały dla wszystkich typów prostych:

* long
* int
* short
* byte
* char
* double
* float
* boolean

**Konstruktor** – jest to specjalna metoda danej klasy, która jest wywoływana podczas tworzenia jej instancji. Podstawowym zadaniem konstruktora jest zainicjowanie obiektu, a w niektórych językach programowania także utworzenie obiektu.

**Zadania konstruktora:**

* obliczanie rozmiaru obiektu.
* alokacja obiektu w pamięci.
* wykonanie kodu wywołanego konstruktora.
* wykonanie kodu klasy bazowej.
* wyczyszczenie (czyli zerowanie) obszaru pamięci zarezerwowanej dla obiektu.