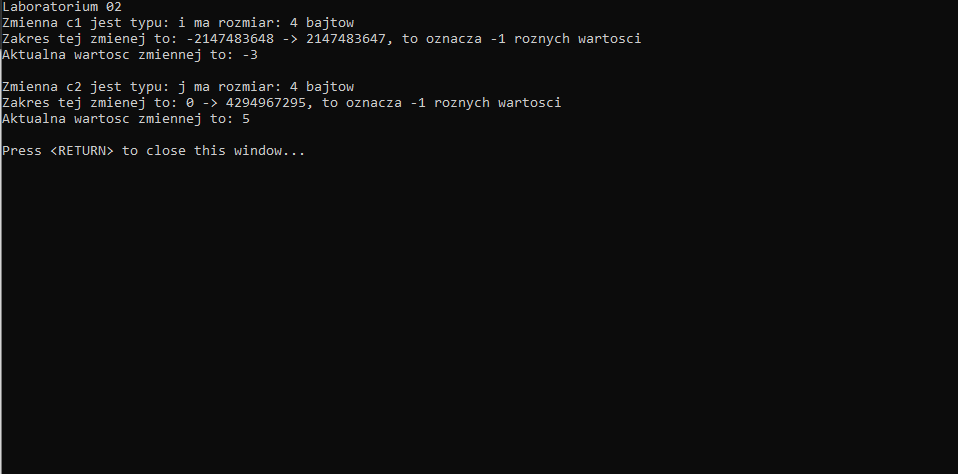
Jakub Łabendowicz 22.03.2020r.

**Sprawozdanie**

# Zadanie 2.3



* Jakie rozszerzenie ma plik nagłówkowy i po co on jest?

.h – deklaracja i definiowanie klasy oraz deklaracja metod klasy

* Jakie rozszerzenie ma plik źródłowy i po co on jest?

.cpp – definiowanie metod klasy

* Listing 2.6, wiersze 1,2,3,4 → Jaki jest efekt dyrektywy include?

Załączenie biblioteki, pliku.

* Co oznacza operator :: (np. w linii 8 na listingu 2.5)?

Metody są statyczne dlatego uruchamiamy je operatorem ::. Jest to odwołanie do pliku nagłówkowego.

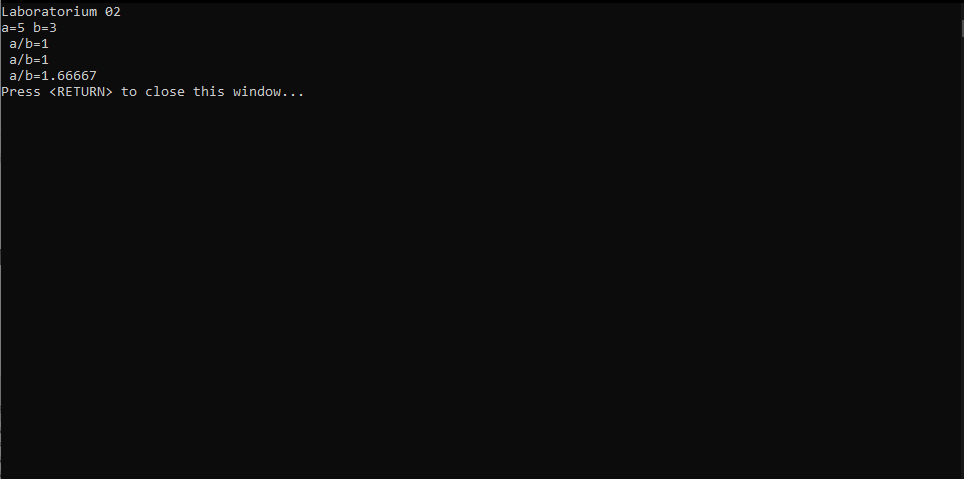
* Listing 2.7, wiersz 15 i 19 → Dlaczego zastosowany został typ long.

Long ma większy zakres.

* Jaki jest efekt słowa endl?

Przejście do nowej linii.

# Zadanie 2.4



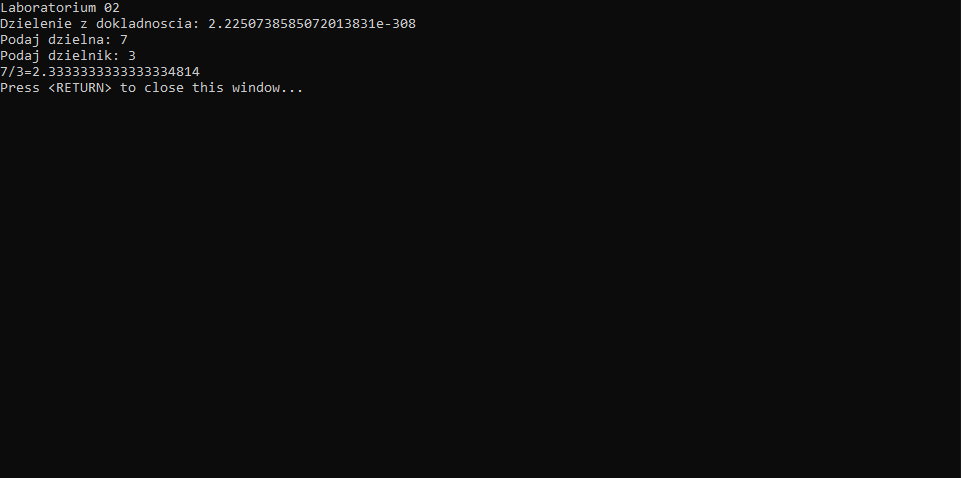
* Dlaczego wyniki dzielenia są różne?

Inny typ zmiennej. Int nie przetrzymuje w pamięci liczb po przecinku.

* Listing 2.9, wiersz 15 → Co tu się dzieje?

Zamiana typu na double, działanie dzielenia oraz przypisanie wyniku do zmiennej.

# Zadanie 2.5



* Zmień metodę zadanie\_2\_5 tak, aby dzielenie wykonywało się bez kontroli dzielnika. Czy działanie programu w tej postaci jest akceptowalne?

Nie, ponieważ wynik przekroczy zakres.

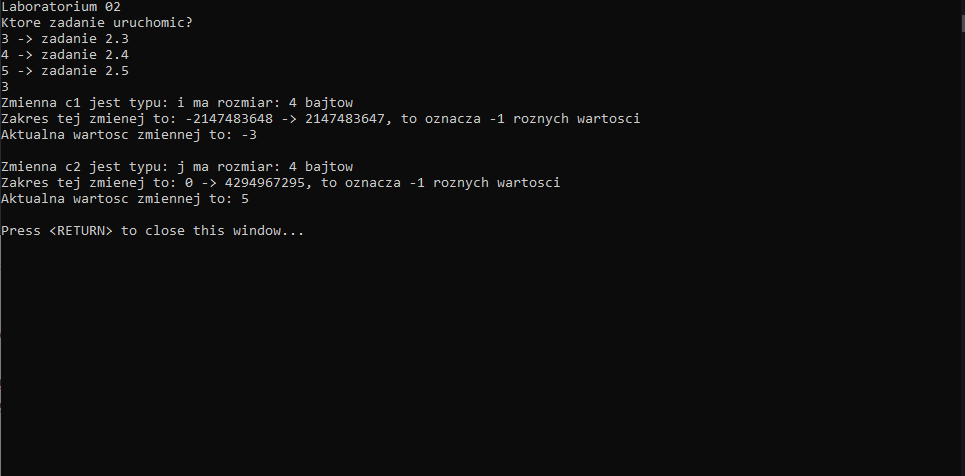
* Opisz co oznacza wartość minimalna typu dla double i dla int .

Każdy typ zmiennych ma własny zakres liczb. Ten zakres może zostać przesunięty, zatem wartość minimalna jest to minimalna liczba jaka może zostać przechowana przez dany typ zmiennej w zależności od przesunięcia zakresu.

* Wyjaśnij czy bezpiecznie jest porównywać liczbę typu double do zera.

Aby porównanie było możliwe zapis zera musi być w postaci 0.0, zatem zero musi być tego samego typu.

# Zadanie 2.6



* Listing 2.12, wiersz 3 → Czy zmienna wybor może mieć typ double?

Tak, lecz może powodować to błędy.

* Listing 2.13, wiersz 3 → Czy zmienna wybor może mieć typ int?

Tak.

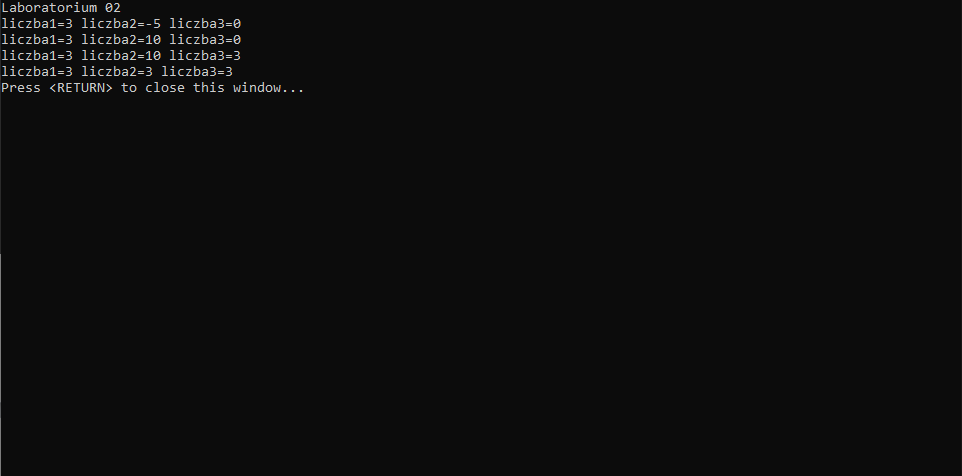
* Listing 2.13, wiersz 11 → Co oznacza II ?

Iloczyn logiczny czyli „lub”.

* Listing 2.13, wiersze 14, 15, 16 → Jaka jest rola słowa break?

Przerywa wykonywanie polecenia.

# Zadanie 2.7



* Listing 2.14, wiersz 6 → W jakim celu wskaźnikowi nadaje się wartość nullptr?

„nullptr”, oznacza wskaźnik pusty, powoduje to zwolnienie pamięci.

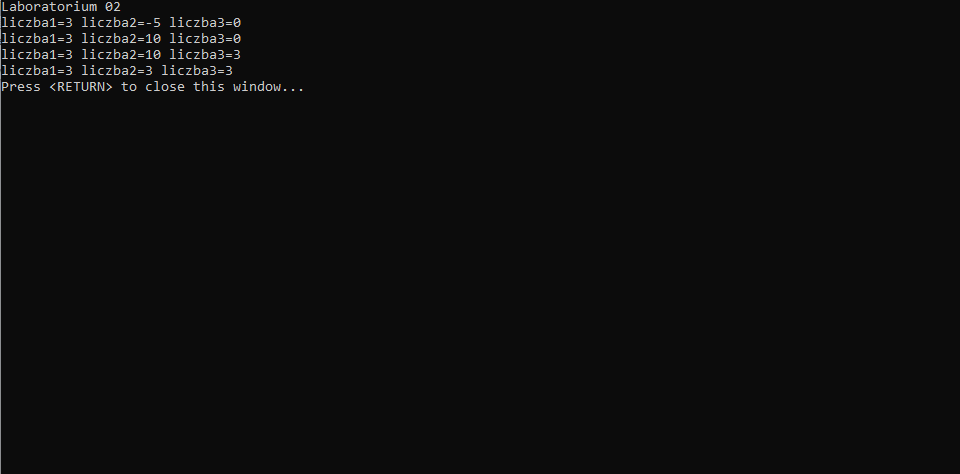
* Listing 2.14, wiersz 12 → Jak działa operator & ?

Odwołanie do lokalizacji w pamięci zmiennej.

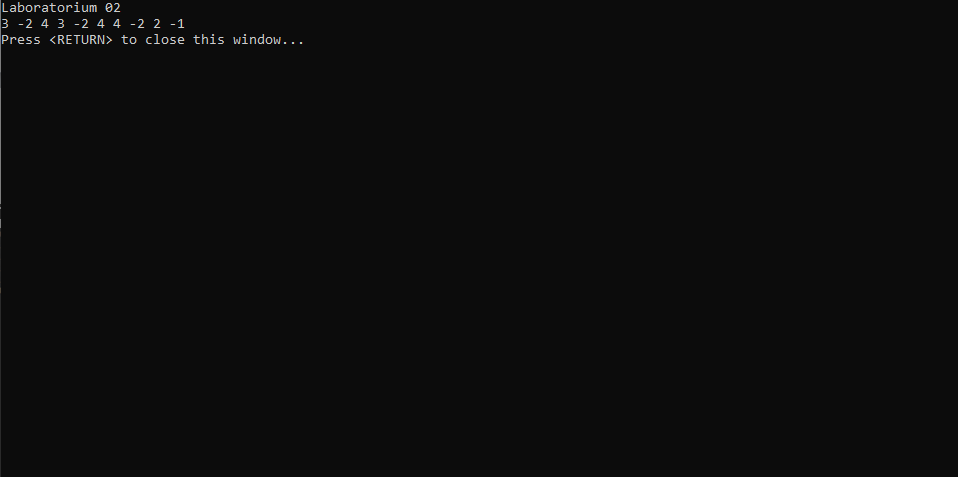
* Listing 2.14, wiersz 14 → Po co jest znak \* ?

Odwołanie do wartości zmiennej wskaźnika.

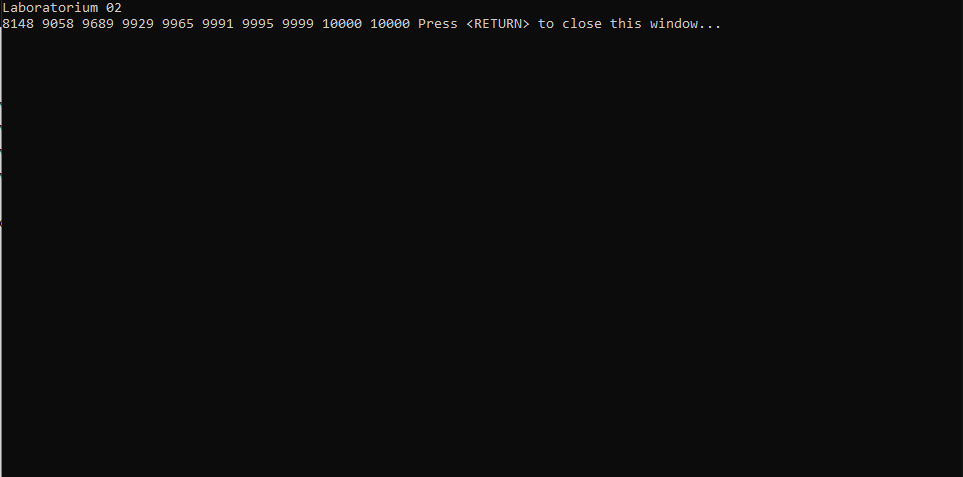
# Zadanie 2.8



# Zadanie 2.9



# Zadanie 2.10



* Listing 2.17, wiersz 4 → Jaki będzie efekt wywołania instrukcji?

Utworzenie tablicy.

* Listing 2.17, wiersz 7 → Co tu się dzieje?

Rzutowanie liczb typu short na podany zakres.

* Listing 2.16, wiersz 12 → Jakie instrukcje znajdują się w bloku pętli while ?

Losowanie kolejnych liczb do momentu, aż nie będą mniejsze od poprzedniej.

* Listing 2.16, wiersz 12 → Jak w ogóle działa pętla while ?

Szuka pierwszego przypadku spełniającego warunek.

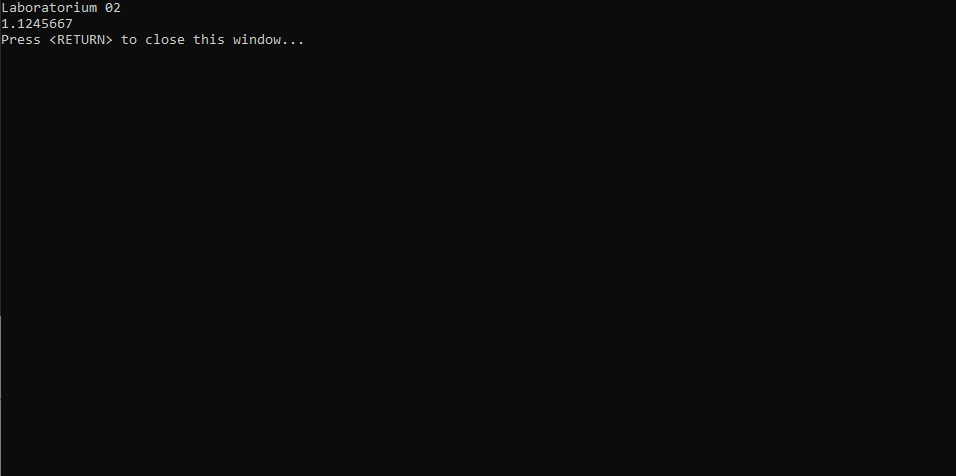
* Czy może zdarzyć się, że warunek zakończenia pętli while nie wystąpi?

Tak.

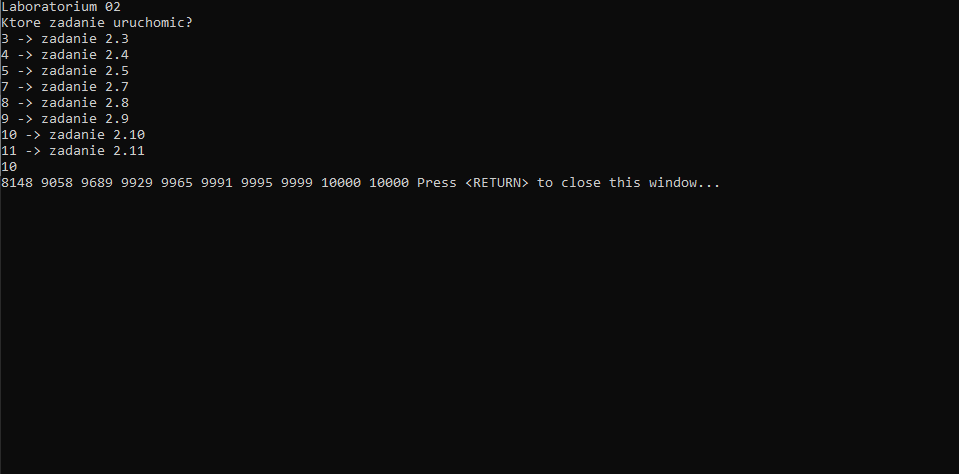
* Czy zaproponowany algorytm można zastosować do wygenerowania ciągu rosnącego?

Tak, ale musiałby „while” posiadać warunek „<=”.

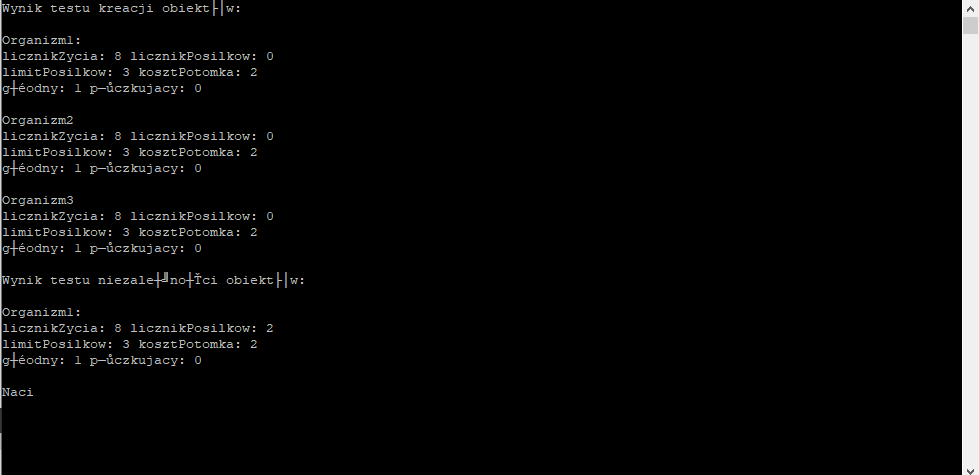
# Zadanie 2.11



# Zadanie 2.12



# Zadanie 3.1



* Listing 3.2 wiersz 10 →Jakie wartości można podstawić pod zmienną nazwa?

Łańcuchy znaków.

* Listing 3.2, wiersz 11 →Czy w funkcji main można podstawić pod zmienną bok1 wartość np. poleceniem obiektKlasy.bok1 = 3 ?

Nie, gdyż argument jest prywatny.

* Listing 3.2, wiersz 15 → Jaką wartość zwraca metoda czyPoprawny?

True lub false.

* Listing 3.2, wiersz 16 → Czy w funkcji main można wywołać metodę obliczObwod poleceniem obiektKlasy.obliczObwod()?

Nie, gdyż metoda jest prywatna.

* Listing 3.2, wiersz 20 → Jaki jest efekt użycia domyślnych wartości argumentów metody?

Użyte zostaną domyślne wartości argumentów, nie musimy nadawać innych wartości.

* Listing 3.2, wiersz 15 oraz wiersz 22 → Czym różnią się metody? Jak nazywa się taki mechanizm?

W pierwszej metodzie musimy nadać dwie wartości. W drugiej nie nadajemy żadnych wartości. Lista inicjalizacyjna konstruktora.

* Listing 3.2, wiersz 23 → Co zwraca metoda?

Łańcuch znaków.

* Listing 3.2, wiersz 28 → Co oznacza std:: ? Czy jest to konieczne?

Przestrzeń nazw, nie.

* Listing 3.3, wiersz 1 → Co oznacza zapis Prostokat:: i po co on jest?

Wywołanie metody przy odwołaniu do klasy.

* Listing 3.3, wiersz 3 → Skąd bierze się zwracana przez metodę wartość?

Z wartości argumentów ustanowionych przy konstruowaniu obiektu.

* Listing 3.6, wiersz 2 → co oznacza zapis :nazwa(n) i jaki jest jego efekt? Czy można zastąpić go przypisaniem nazwa = n w bloku { }?

Jest to instrukcja inicjalizująca, która może inicjalizować pola klasy zaraz po utworzeniu klasy. Tak, lecz będzie trzeba od razu podać wartość.

* Listing 3.6, wiersz 12 → Jaką wartość będzie miało pole bok2?

0

* Listing 3.7 → Czy użycie std:: jest tutaj konieczne?

Tak.

* Listing 3.12 → Jak działa metoda? Dlaczego tutaj można wywoływać metody prywatne klasy?

Jest to funkcja zawarta w klasie. Wywołanie występuje wewnątrz klasy.

* Listing 3.13, wiersz 6, 7, 9 → Co robi funkcja biblioteczna to\_string?

Konwertuje zmienną do „int”.

# Zadanie 3.2

* Listing 3.14, wiersz 14,15,16 → W jaki sposób uzyskuje się dostęp do pola lub metody obiektu mając jego nazwę?

Obiekt.metoda

* Listing 3.14, wiersz 9 → Jakie wartości przyjmuje konstruktor?

Domyślne.

* Listing 3.14, wiersz 13 → Co robi metoda precision wywołana na obiekcie automatycznym cout?

Podaje ilość miejsc po przecinku.

* Listing 3.14, wiersze 14, 15, 16 → Dlaczego efekt metody precision nie dotyczy tych poleceń?

Zwracana wartość jest typu string.

* Listing 3.14, wiersze 21-25 → Ile instrukcji języka C++ jest w tych wierszach?

2

* Wyjaśnij czy istnieje możliwość zmiany nazwy obiektu klasy prostokąt, np.: p1.nazwa = Inna nazwa ”Inna nazwa” ”Inna nazwa”?

Nie.

* Listing 3.14 → W definicji metody nie ma wywołania destruktora obiektu. Czy destruktor się uruchamia?

Tak.

# Zadanie 3.3

* Co oznacza, że obiekt klasy organizm jest „pączkujący”. Znajdź odpowiednie fragmenty w kodzie.

Znaczy że jest żywy i ilość zjedzonych przez niego posiłków jest większa od kosztu potomka, umożliwia to stworzenie potomka, po jego stworzeniu ilość posiłków rodzica spada o koszt potomka.

* Listing 3.15, wiersz 4 i 5 → Jaki efekt ma słowo const ?. Czy różne obiekty klasy Organizm mogą mieć inne wartości pola kosztPotomka ? Czy wartość pola limitPosi kow łkow może być odczytana w funkcji zewnętrznej np. w main ?

Po stworzeniu obiektu nie będzie możliwości zmiany tych wartości. Tak. Tak.

* Listing 3.15, wiersz 8, 9 → Dlaczego pola licznikZycia oraz licznikPosilkow nie mogą być const ?

Wartości te zmieniają w trakcie życia obiektu np.: po stworzeniu potomka.

* Listing 3.16, wiersz 16 → Co oznacza słowo const dołączone do nazwy metody ?

Deklaruje, że dana zmienna nie zmienia stanu obiektu.

* Listing 3.16 → Które metody będzie można wywołać na obiekcie klasy Organizm ?

Wszystkie publiczne zdefiniowane w klasie obiektu.

* Listing 3.17 → Które metody będzie można wywołać na stałej klasy Organizm, tzn. obiekcie zdefiniowanym np. const Organizma organizm(10,4,2); ?

Niezmieniające wartości tj. zadeklarowne jako const.

* Czy można stworzyć obiekt klasy Organizm instrukcją: Organizm organizmPusty; ?

Nie.

* Listing 3.16, wiersze 4,5,6,7 → Co tutaj się dziej? Czy można alternatywnie zastąpić te wiersze podstawieniami w bloku { } ?

Następuje zapis podanych danych do prywatnych zmiennych obiektu, w wierszu 7 następuje ustawienie licznika posilkow na zero. Podsumowując następuje wpisanie do zmiennych właściwości organizmu.

* Listing 3.17 → Co robi metoda posilek ? Tzn. jaki jest jej algorytm?

Sprawdza czy organizm może coś zjeść, jeśli tak zwiększa liczbę zjedzonych posiłków i zwraca true, jeśli nie zwraca false.

* Listing 3.18 → Co robi metoda potomek ?

Sprawdza czy organizm spełnia warunki potrzebne do pączkowania, jeśli tak zmniejsza licznik posiłków o koszt potomka i zwraca true, jeśli nie zwraca false.

* Listing 3.19 → Co robi metoda krokSymulacji ?

Zlicza wiek organizmu jeśli ten jest żywy.

# Zadanie 3.4

* Listing 3.21 → Ile instrukcji języka C++ zostało umieszczonych w wierszach od 3 do 13?

1

* Listing 3.21, wiersz 1 → Co jest parametrem zdefiniowanej funkcji ?

Adres organizmu.

* Listing 3.21 → Które metody klasy Organizm nie mogą być użyte wewnątrz funkcji na obiekcie o ?

Które nie są zdeklarowane jako const.

* Listing 3.22, wiersz 5 → Jaki konstruktor został użyty do stworzenia obiektu organizm2 ? Przy tworzeniu obiektu jest zawsze uruchamiany jakiś konstruktor.

Domyślny konstruktor klasy organizm. Tak.

* Listing 3.22 → Na czym polega test niezależności obiektów ?

Sprawdza czy poszczególne obiekty tej samej klasy mogą się zmieniać nie zależnie.

* Listing 3.22 → Na czym polega test symulacji ?

11 razy wykonuje funkcję krokSymulacji na organizm4, i jeśli może mieć potomka to go tworzy, jeśli nie to wywołuje funkcje posilek obiektu organizm4, po każdym obrocie pętli wyświetla stan obiektu.