Jakub Łabendowicz 24.04.2020r.

Sprawozdanie

LABORATORIUM 7.

# Pytania kontrolne:

## Przedstaw sposób przeciążenia operatora przypisania. (Patrz listing 7.12. oraz 7.14)

Operator zdefiniowany przez użytkownika, polega na zdefiniowaniu własnego operatora przypisania, który usunie wskazywane obszary pamięci i w ich miejsce utworzy nowe oraz skopiuje informacje zawarte w obszarach wskazywanych przez pola – wskaźniki obiektu źródłowego do obszarów wskazywanych przez pola – wskaźniki obiektu docelowego.

# Zadanie 7.1. Stworzenie klas Glon, Grzyb oraz Bakteria

## Listing 7.1, wiersz 4 → Jaki jest efekt słowa final ?

Metoda klasy nie będzie dziedziczona.

## Listing 7.4, wiersz 5 i 6 → Co się tutaj dzieje ?

Przypisanie wartości ustawień symulacji do klasy UST(dziedziczenie) poprzez wywołanie metody pobierzustawienia().

## Listing 7.5, wiersze od 1 do 5 → Wytłumacz jak to działa?

Są tutaj definiowane parametry organizmu na podstawie ustawień symulacji. Ilość życia jest losowana przez metodę losujPomiedzy(), natomiast reszta parametrów tj. LimitPosilkow i Koszt Potomka są przekazywane na sztywno z klasy UstawieniaSymulacji do UST.

## Listing 7.6, wiersz 2 → Czy można zastosować tutaj if ?

Nie.

## Listing 7.7, wiersz 3 → Skąd się wzięła metoda potomek ?

Ponieważ klasa Glon dziedziczy klasę Organizm, a metoda potomek() jest metodą klasy Organizm, dlatego można użyć tej metody odwołując się do klasy Glon.

## Listing 7.8 → Jaka jest różnica w algorytmach metod przyjmijZdobycz ?

Algorytm dla Glona usuwa mieszkańca, jeżeli jego wskaźnik nie jest pusty. Algorytm dla Grzyba sprawdza czy wskaźnik mieszkańca nie jest pusty, sprawdza czy dany Organizm jest żywy na podstawie wartości mieszkanca(0), jeśli jest trupem następuje wywołanie metody posiłek(). Na końcu wskaźnik mieszkaniec jest usuwany. Algorytm dla Bakterii sprawdza czy wskaźnik mieszkańca nie jest pusty, potem przypisuje mu RodzajMieszkanca r na podstawie wartości mieszkanca(0), jęzeli wartość r odpowiada Glonowi lub Grzybowi następuje wywołanie metody posiłek(). Na końcu wskaźnik mieszkaniec jest usuwany

## Listing 7.9 → Opisz algorytm metody.

Metoda ta symuluje i sprawdza stany dla każdego rodzaju organizmu. Jeśli dany organizm jest żywy to licznik życia organizmu zostaje zmniejszony o 1, następnie metoda sprawdza czy organizm jest żywy i pączkujący i czy posiada sąsiedztwo wtedy zwraca zamiar jako potomek wraz z jego położeniem i kończy algorytm. Jeśli powyższy warunek nie jest spełniony to algorytm sprawdza czy organizm sprawdza czy jest żywy i głodny, jeśli tak wywołuje metodę posiłek(), która zwiększa ilość posiłków. Następnie algorytm sprawdza czy dany organizm jest martwy i jego symbol nie jest ustawiony jako TRUP, wtedy symbol mieszkańca zostaje zmieniony na znakTRUP i metoda zwraca ZamiarMieszkanca z nowymi parametrami.

## Listing 7.10 → Opisz algorytm metody.

Pierwsze 2 kroki metody tj. powyżej(„…położeniem i kończy algorytm.”). Następnie algorytm sprawdza czy organizm jest żywy i głodny i czy posiada w swoim sąsiedztwie TRUPA, jeśli tak metoda zwraca ZamiarMieszkanca z akcją ROZKLAD i nowym położeniem. Następnie algorytm sprawdza czy dany organizm jest martwy i jego symbol nie jest ustawiony jako TRUP, wtedy symbol mieszkańca zostaje zmieniony na znakTRUP i metoda zwraca ZamiarMieszkanca z nowymi parametrami.

## Listing 7.11 → Opisz algorytm metody.

Pierwsze 2 kroki metody tj. powyżej(„…położeniem i kończy algorytm.”). Następnie algorytm sprawdza czy organizm jest żywy i głodny oraz czy w swoim sąsiedztwie posiada GLON i zwraca ZamiarMieszkanca z akją POLOWANIE i sąsiedztwem GLON. Potem algorytm sprawdza czy organizm jest żywy i głodny oraz czy w swoim sąsiedztwie posiada BAKTERIA i zwraca ZamiarMieszkanca z akją POLOWANIE i sąsiedztwem BAKTERIA. Następnie algorytm sprawdza czy dany organizm jest martwy i jego symbol nie jest ustawiony jako TRUP, wtedy symbol mieszkańca zostaje zmieniony na znakTRUP i metoda zwraca ZamiarMieszkanca z nowymi parametrami.

# Zadanie 7.2 Stworzenie klasy Nisza

## Listing 7.12, wiersz 4 → Czy zamiast wskaźnika można użyć referencji ?

Nie.

## Listing 7.12, wiersz 17 → Jak to działa ?

Podczas wywoływania metody zwraca stałą wartość wskaźnika, która nie może być pusta.

## Listing 7.12, wiersz 23 i 25 → Jakie konsekwencje pociąga za sobą słowo const ?

Wartość jest stała i może się zmienić tylko podczas kolejnego wywołania danej metody

## Listing 7.12, wiersz 40 → Dlaczego użyto symbolu -> a nie kropki ?

Zmienna jest wskaźnikiem.

## Listing 7.13 → Czy operator kopiujący mógłby mieć nagłówek: Nisza::Nisza(const Nisza & innaNisza) ? Czy to jest typowy operator kopiujący?

Nie. Gdy typ argumentu konstruktora kopiującego nie jest wartością stałą, inicjowanie przez skopiowanie obiektu const generuje błąd. Odwrócenie nie jest prawdziwe: Jeśli argument jest stałą, można zainicjować przez skopiowanie obiektu, który nie jest stałą.

## Listing 7.14 → Czy operator przypisania mógłby mieć nagłówek: Nisza &Nisza::oper

Nie.

# Zadanie 7.3. Test klas Mieszkaniec oraz Nisza

## Listing 7.17, wiersz 7 → Czy słowo static jest konieczne ?

Nie.

## Listing 7.17, wiersz 30 → Jak to działa ?

Następuje tutaj przeciążenie operatora przypisania. Podczas tego procesu obiekt pod zmienną n2 zostaje podmieniony z obiektem pod zmienną n1, poprzez wykorzystanie wskaźnika do obiektu tmp(plik nisza.cpp).

# Zadanie 7.4. Test określania sąsiedztwa

## Czy metoda aktywujLokatora działa poprawnie?

Nie, trzeba zmienić typ metody na public.