Programowanie obiektowe

Zajęcia 1. Środowisko, podstawowe pojęcia, enkapsulacja

Zadanie 1. Przygotowanie środowiska

1. Zainstaluj Visual Studio Code odpowiednie dla Twojego systemu operacyjnego (https://code.visualstudio.com/Download).
2. Uruchom środowisko.
3. Wejdź w sekcję rozszerzenia i wyszukaj a następnie zainstaluj rozszerzenie `C/C++ IntelliSense, debugging, and code browsing` firmy Microsoft (daje radę również w innych systemach).
4. Zainstaluj pakiet narzędzi do kompilacji C++ zgodny z Twoim systemem operacyjnym
   - dla systemu Windows przykładowo :
     - MinGW 64
     - Microsoft Build Tools (zakładka Tools for Visual Studio 2019->Build Tools for Visual Studio 2019)
   - dla systemu Ubuntu przykładowo:
     - `sudo apt-get install build-essential`
   - dla systemu MacOS przykładowo:
     - llvm np. przy pomocy brew. Do tego celu możesz zainstalować brew korzystając z instrukcji na stronie: https://brew.sh/. Następnie żeby zainstalować llvm skorzystaj z komendy: `brew install llvm`

5. Upewnij się, że pakiet narzędzi do kompilacji jest dostępny w zmiennej PATH (np. czy można uruchamiać w konsoli kompilator)

6. Zainstaluj pakiet narzędzi rozszerzających do programowania w języku Java (odpowiedni dla Twojego systemu operacyjnego: Windows, MacOS) lub zainstaluj odpowiednie narzędzia i rozszerzenia VSC jeśli Twój system operacyjny nie jest dostępny (Java Development Kit, rozszerzenia w VSC do debugowania i kompilacji w języku Java).

Zadanie 2. Projekt "witaj świecie" w C++

1. W wybranym miejscu na dysku utwórz folder o nazwie "HelloCpp".

2. W środowisku Visual Studio Code otwórz folder (File->Open Folder...).

3. Wybierz do otworzenia wcześniej utworzony folder.

4. Dodaj do folderu plik `hello.cpp`.

5. Umieść w pliku następujący kod:

```
#include <iostream>
#include <string>

int main() {
    std::string fellow = "world";
    std::cout << "Hello " << fellow << std::endl;
    while (fellow != "exit") {
        std::cout << "Introduce yourself: ";
        std::cout.flush();
        std::getline(std::cin, fellow);
        std::cout << "Hello " << fellow << std::endl;
    }
}
```

6. Utwórz punkt przerwania programu (breakpoint) w linii deklaracji zmiennej fellow.

7. Uruchom program w trybie debug (np. poprzez naciśnięcie przycisku F5).

8. Wybierz konfigurację, za pomocą której będziesz debugować program (zależnie od kompilatora i systemu operacyjnego będą do wyboru inne konfiguracje).

9. Przejdź przez program krok po kroku wpisując kolejno imiona trzech swoich ulubionych postaci z książek/komiksów/filmów a następnie wpisz "exit".

10. Opisz krótko swoje obserwacje z debugowania kodu.

11. Zmodyfikuj kod tak żeby w przypadku wpisania wartości "exit" program nie wypisywał "Hello exit".

Zadanie 3. Projekt "witaj świecie" w Java'ie

1. W wybranym miejscu na dysku utwórz folder o nazwie "HelloJava"

2. W środowisku Visual Studio Code otwórz folder (File->Open Folder...).

3. Wybierz do otworzenia wcześniej utworzony folder.

4. Dodaj do folderu projektu pakiet "hello".

5. Do pakietu "hello" dodaj implementacja klasy World z następującym kodem:

```
package hello;

import java.util.Scanner;

public class World {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        String fellow = "World";
        System.out.printf("Hello %s\n", fellow);
        while (fellow != "exit") {
            System.out.printf("Introduce yourself: ");
            fellow = s.nextLine();
            System.out.printf("Hello %s\n", fellow);
        }
        s.close();
    }
}
```

6. Skompiluj program i uruchom go w trybie debug (np. poprzez wciśnięcie F5).

7. Wybierz kofiguracją kompilacji i uruchamiania w trybie debug.

8. Czy program działa zgodnie z oczekiwaniem? Na czym polega problem?

9. Zdebuguj program i określ przyczynę problemu następnie zmodyfikuj program tak by nie występował.

Zadanie 4. Projekt enkapsulacja (w C++ albo Java)

Korzystając z VSC napisz program, w którym zamodelowane będzie działanie jednego z poniższych bytów. To, który spośród bytów powinieneś modelować powinno zależeć od następującego wzoru:

$$numer\_indeksu \ \mathbf{mod}\ 4 + 1$$

W ramach projektu utwórz instancję zaimplementowanego bytu i przetestuj jej działanie.

1. `Ul`, którego cechują następujące własności:

   - liczba pszczół,
   - położenie ula (we współrzędnych geograficznych),
   - średni roczny wskaźnik ilości produkowanego miodu,
   - nazwa pasieki,
   - (zaproponuj własną)

   Zadbaj by stan obiektu był zabezpieczony zgodnie z założeniami enkapsulacji i można było do niego się dostać tylko i wyłącznie za pomocą dedykowanych metod:

   - ustawienia startowej liczby pszczół (metoda powinna ustawiać wartość tylko raz),
   - określenia wartości zmiany liczby pszczół w przeciągu zadanego okresu przy czym zmiana nie powinna przekraczać zakresu $\langle -100, 100 \rangle$ a sama ilość pszczół w ulu nie powinna być mniejsza od 0,
   - odczytania aktualnej liczby pszczół,
   - określania położenia ula (weryfikujące poprawność odpowiednich koordynat),
   - odczytania aktualnej pozycji ula,
   - dodania aktualnej rocznej ilości wyprodukowanego przez ul miodu,
   - odczytywania średniego rocznego wskaźnika ilości produkowanego miodu (przed pierwszym dodaniem rocznej ilości wyprodukowanego miodu powinna być zwracana wartość 0),

- określenia nazwy pasieki (nazwa powinna być zapisana wtedy i tylko wtedy gdy rozpoczyna się od dużej litery),
- odczytania nazwy pasieki
- (zaproponuj własne metody dotyczące wcześniej zaproponowanej własności)

2. `ElektrowaniaWeglowa`, którą cechują następujące własności:

   - maksymalna ilość węgla w podajniku,
   - stan magazynowy węgla,
   - sprawność elektrowni,
   - liczba pracowników,
   - (zaproponuj własną)

   Zadbaj by stan obiektu był zabezpieczony zgodnie z założeniami enkapsulacji i można było do niego się dostać tylko i wyłącznie za pomocą dedykowanych metod:

   - ustawienia maksymalnej ilości węgla, który może obsługiwać podajnik (metoda powinna zadziałać tylko raz),
   - odczytania maksymalnej ilości węgla w podajniku,
   - określenia wartości ilości węgla dodanej do magazynu,
   - określenia, iż przerzucono z magazynu do podajnika węgiel w ilości maksymalnej ilości węgla obsługiwanej przez podajnik (chyba, że stan magazynowy nie pozwala na taką operację),
   - odczytania aktualnej ilości węgla w magazynie,
   - określenia wartości sprawności elektrowni (podanej w postaci współczynnika o wartościach z przedziału $(0, 1\rangle$),
   - odczytania wartości sprawności elektrowni,
   - określenia liczby przyjętych pracowników,
   - określenia liczby pracowników odchodzących z pracy (zabezpieczyć żeby wartość nie spadła poniżej 0),
   - odczytania aktualnej liczby pracowników,
   - (zaproponuj własne metody dotyczące wcześniej zaproponowanej własności)

3. `RadaNadzorcza`, którą cechują następujące własności:

- liczba członków,
- średnia liczba wydanych w roku uchwał,
- aktualny budżet rady,
- data ostatniego zgromadzenia,
- (zaproponuj własną)

Zadbaj by stan obiektu był zabezpieczony zgodnie z założeniami enkapsulacji i można było do niego się dostać tylko i wyłącznie za pomocą dedykowanych metod:

- określenia liczby członków powołanych przez walne zgromadzenie,
- określenia liczby członków odwołanych przez walne zgromadzenie (zadbaj by nie było możliwości zmniejszenia liczby członków rady poniżej 0),
- odczytania aktualnej liczby członków rady nadzorczej,
- określenia liczby uchwał wydanych w poprzednim roku,
- odczytania średniej liczby uchwał ze wszystkich lat działania rady,
- określenia wpływu do budżetu,
- określenia wydatków z budżetu,
- odczytania aktualnego stanu budżetu,
- określenia daty planowanego zgromadzenia rady (data ta musi być datą z przyszłości, przy czym może być tylko jedno planowane zgromadzenie rady, które nie przerodziło się jeszcze w datę odbytego zgromadzenia) data ta powinna nadpisywać datę ostatniego zgromadzenia w momencie któregokolwiek odwołania się do dat zgromadzenia (czy to planowanego czy ostatniego),
- odczytania daty ostatniego zgromadzenia,
- (zaproponuj własne metody dotyczące wcześniej zaproponowanej własności)

4. `JednorekiBandyta`, którego cechują następujące własności:

   - koszt pojedynczej gry,
   - procent kosztu przechodzący do puli nagród,
   - prawdopodobieństwo wygranej,
   - pula nagród,

- liczba rozegranych gier,
- (zaproponuj własną)

Zadbaj by stan obiektu był zabezpieczony zgodnie z założeniami enkapsulacji i można było do niego się dostać tylko i wyłącznie za pomocą dedykowanych metod:

- określenia kosztu pojedynczej gry (koszt musi być większy od 0),
- odczytania kosztu pojedynczej gry,
- określenia procentu kosztu przechodzącego do puli nagród (liczba z zakresu od 1 do 70),
- określenie prawdopodobieństwa wygranej (wartość z przedziału $\langle 0.1, 0.2 \rangle$),
- odczytania procentu kosztu przechodzącego do puli nagród,
- przeprowadzenie gry, które odpowiada na pytanie czy wygrano i jeśli tak to jaką kwotę (dla ułatwienia jeśli się wygrywa to zawsze całość puli),
- odczytanie aktualnej puli nagród,
- odczytanie liczby rozegranych gier,
- (zaproponuj własne metody dotyczące wcześniej zaproponowanej własności)