Opracowanie: dr inż. Łukasz Maliński, Politechnika Śląska, 2022

LABORATORIUM 2.1
KLASY I ENKAPSULACJA

Zadanie podstawowe [2,0 pkt.]

■ Zdefiniuj klasę do opisu **Przedmiotu** (w znaczeniu zajęć na uczelni). Klasa powinna przechowywać informację takie jak:

- nazwa przedmiotu,
- wskazanie nauczyciela prowadzącego (inny typ złożony w programie),
- liczba godzin (jedno pole) z podziałem na wykłady, ćwiczenia i laboratoria,
- liczba punktów ECTS,

Samodzielnie dobierz właściwe typy pól.

Wskazówki: Zauważ, że liczba godzin to informacja złożona z 3 uporządkowanych liczb. Zastanów jaki typ danych dobrać, aby 3 liczby tego samego typu zapisać pod jednym polem klasy.

■ Dla klasy **Przedmiot** zdefiniuj akcesory uwzględniające następujące ograniczenia:

- żadna z liczb w polach klasy nie może być ujemna,
- liczebności godzin muszą być wielokrotnością liczby 15,
- przedmiot nie może mieć więcej niż 10 pkt. ECTS.

Ponadto geter i seter liczebności godzin powinien zwracać/ustawiać liczbę godzin dla wskazanego typu zajęć, do wskazywania typu zajęć użyj typu wyliczeniowego.

Wskazówki: Jeśli podane wartości nie spełniają ograniczeń zawsze zastępuj je najbliższą poprawną wartością. W tym przypadku, zarówno geter jak i seter powinny przyjmować dodatkową informację – typ zajęć. Definiując typ wyliczeniowy dla typu zajęć pamiętaj o przypisaniu klucza liczbowego do wartości – ułatwi to indeksowanie wartości.

■ Zdefiniuj metodę użytkową pozwalającą wyliczyć ile godzin student powinien poświecić na samodzielną pracę w domu przyjmując, że 30 godzin pracy to jest 1 punkt ECTS, a liczebności godzin dotyczą tylko zajęć kontaktowych (realizowanych na uczelni, z udziałem nauczyciela akademickiego).

Wskazówki: Sumaryczna liczba godzin kontaktowych podzielona przez 30 to liczba ECTS realizowana kontaktowo. Pozostałe punktu (brakujące do pełnej liczby punktów za przedmiot) pomnożone przez 30 to liczba godzin jaką przeciętny student powinien poświecić na pracę w domu.

■ Zademonstruj działanie klasy w programie głównym definiując dwa rożne przedmioty w tym PK i jeden inny przedmiot (pobierz autentyczne dane ze strony wydziału). Koniecznie użyj metody użytkowej i wyświetl wynik na ekranie – wyciągnij wnioski!

Ocenianie:

Dobór typów pół klasy:	0,4 pkt.
Definicje akcesorów:	0,6 pkt.
Metoda użytkowa:	0,4 pkt.
Prezentacja w programie głównym:	0,4 pkt.
Ogólna jakość kodu:	0,2 pkt.

Zadanie ambitne [2,0 pkt.]:

■ Zdefiniuj klasę **Przycisk** (potencjalny element GUI), która przechowywać będzie informacje o:

- wymiarach geometrycznych na ekranie (w pikselach),
- współrzędnych położenia lewego górnego rogu przycisku na ekranie,
- stanie aktywności przycisku (aktywny/nieaktywny),

- adresie funkcji globalnej postaci void funkcja(void), która ma być wywołana w chwili naciśnięcia przycisku,
- napisu wyświetlanego w centrum przycisku.

Samodzielnie dobierz typy pól.

Uwaga: Zadanie nie obejmuje tworzenia reprezentacji graficznej Przycisku ani pisania funkcjonalnego GUI. Celem jest tylko modelowanie działania Przycisku w tle!

Wskazówki: Traktuj piksel jako niepodzielną jednostkę. Zastanów się, czy do opisu stanu lepiej użyć typu logicznego, czy wyliczeniowego. Przypomnij sobie jak można przechowywać adres funkcji (podobne zadanie było już wcześniej). Stan aktywności oznacza, czy przycisk da się w danej kwili nacisnąć. Może on być zmieniany na przykład, gdy wyzwalana nim funkcjonalność programu w danej chwili nie powinna być dostępna, ale przycisk powinien dalej istnieć i być widoczny w GUI.

■ Dla klasy **Przycisk** zdefiniuj podstawowy interfejs zawierający akcesory do wszystkich pól klasy. Przy definiowaniu seterów wprowadź ograniczenia tam gdzie ma to sens. Ograniczenia dobierz samodzielnie na bazie opisu klas.

Wskazówki: Przyjmij, że piksele na ekranie mają współrzędne od (0,0) do wymiarów ekranu określonych rozdzielczością. Rozdzielczość ekranu można sobie założyć dowolną i przechować dla uproszczenia w jakieś globalnej stałej. Przyjmij że rozmiar tekstu ma stałe górne ograniczenie ustalone na etapie kompilacji.

■ Stosując się do zasady DRY, napisz jeden seter zbiorczy pozwalający ustawić cały stan instancji (wszystkie pola) równocześnie.

Wskazówki: Staraj się nie kopiować kodu i korzystać z napisanych wcześniej funkcjonalności.

■ Rozbuduj/przebuduj interfejs klasy, tak aby uwzględniał następujące wytyczne:

- Przyjmij, że przycisk nie może być aktywny jeśli nie ma przypisanej funkcji do wykonania. Usunięcie adresu funkcji, powinno natychmiast go dezaktywować, ale ustawienie konkretnej funkcji nie powinno automatycznie go aktywować.

  Wskazówki: Usunięcie adresu funkcji to „ustawienie jej na nullptr”.

- Sprawdzenie ograniczeń dla wymiarów przycisku i współrzędnych ekranowych zdefiniuj w seterach prywatnych, wywoływanych przez setery publiczne.

  Wskazówki: Zauważ, że sprawdzenie wymiaru w pionie i poziomie to ten sam kod różniący się tylko wartościami progowymi. Spróbuj napisać go tak, aby znalazł się w jednej metodzie prywatnej, której można wskazać, na jakich wartościach progowych ma pracować. Metodę tą wywołuj w seterach publicznych.

■ Napisz metodę clicked(), która jeśli przycisk jest aktywny to wywoła przypisaną do niego funkcję globalną w programie.

Wskazówki: Zauważ, że docelowa funkcja nie przyjmuje argumentów, ani nie zwraca wyniku.

■ Zademonstruj działanie klasy w programie głównym.

*Wskazówki: Na potrzeby napisz prostą funkcję, która tylko wypisze jakiś komunikat na ekranie (użycie cout jest tu dozwolone), przypisz ją do instancji **Przycisku** i sprawdź, czy zostanie wywołana gdy dla tej instancji wywołamy metodę clicked().*

Ocenianie:

Dobór typów pól:	0,4 pkt.
Definicje akcesorów:	0,6 pkt.
Realizacja zasady DRY:	0,4 pkt.
Prezentacja metody clicked():	0,4 pkt.
Ogólna jakość kodu:	0,2 pkt.