Nazwa kwalifikacji: **Projektowanie, programowanie i testowanie aplikacji**

Oznaczenie kwalifikacji: **INF.04**

Numer zadania: **02**

Kod arkusza: **INF.04-02-23.01-SG**

Wersja arkusza: **SG**

Lp.	Elementy podlegające ocenie/kryteria oceny
R.1	**Rezultat 1: Implementacja, kompilacja, uruchomienie programu**
	Uwaga: kryteria należy odnieść do aplikacji konsolowej, jeżeli ta nie istnieje, zastosować 1.1 ÷ 1.6 do aplikacji mobilnej *Wystarczy, że sprawdzaną cechę zastosowano dla większości przypadków w kodzie*
R.1.1	Kod źródłowy zapisano w sposób czytelny: instrukcje w osobnych liniach, stosowane spacje pomiędzy operatorami, konsekwentnie stosowana wybrana konwencja dla nawiasów klamrowych
R.1.2	Kod zapisano z wcięciami dla zagłębień bloków
R.1.3	Użyto znaczące nazewnictwo metod / funkcji
R.1.4	Użyto znaczące nazewnictwo zmiennych / pól oraz klasy. Wyjątkami od reguły są zmienne bufor, tmp, iteratory pętli itp. Kryterium **nie jest** spełnione tylko wtedy, gdy nazwy zmiennych nic nie znaczą, np. x, fun
R.1.5	Zastosowano typy zmiennych pasujące do problemu (np. dowolny typ numeryczny dla identyfikatora i licznika; typ napisowy dla tytułu i treści notatki; dopuszcza się w języku Python bez typu)
R.1.6	Podjęto próbę skompilowania kodu, co udokumentowano zrzutem ekranowym przedstawiającym uruchomiony program lub jego kompilację
R.1.7	Program nawiązuje zrozumiałą komunikację z użytkownikiem. Wyświetla tytuł i treść notatki. Jeżeli kod nie uruchamia się z powodu błędów kompilacji - sprawdzić w kodzie aplikacji
R.2	**Rezultat 2: Aplikacja konsolowa**
	Uwaga: kryteria 2.1 ÷ 2.7 należy sprawdzić w kodzie programu, sprawdzane elementy muszą być zapisane zgodnie ze składnią. *Gdy aplikacja nie uruchamia się, a zdający zapisał zrzuty ekranu z uruchomienia aplikacji należy sprawdzić powód braku kompilacji. Jeśli występują błędy w plikach źródłowych zdającego kryteria 2.8 i 2.9 nie są spełnione. Jeżeli błędy występują w innych plikach ocenić na podstawie kodu i zrzutu ekranu* *W kryteriach 2.5 ÷ 2.7 dopuszcza się funkcje zamiast metod (podejście strukturalne)*
R.2.1	Kod składa się z programu głównego oraz definicji klasy *notatka* w której zdefiniowano przynajmniej jedno pole i przynajmniej jedną metodę zgodne z treścią zadania (może być niedokończona, lub z błędami)
R.2.2	Klasa zawiera dwa pola numeryczne o zasięgu private oraz dwa pola napisowe o zasięgu protected (w Python zgodnie z konwencją zastosowano dla protected jeden podkreślnik w nazwie, dla private dwa podkreślniki)
R.2.3	Pole licznika notatek jest statyczne oraz jest inkrementowane (lub zwiększane o jeden) w konstruktorze przed przypisaniem jego wartości do pola identyfikatora
R.2.4	Klasa zawiera konstruktor z dwoma parametrami wejściowymi typu napisowego. Parametry są przypisywane do tytułu i treści notatki, do pola identyfikatora jest przypisana wartość licznika
R.2.5	Zdefiniowano dwie metody bezparametrowe oraz nie zwracające wartości (np. typ void). Obie metody mają zakres public
R.2.6	Jedna metoda wyświetla jedynie tytuł i treść notatki
R.2.7	Druga metoda wypisuje zawartość wszystkich pól klasy oddzielonych od siebie średnikiem
R.2.8	Program kompiluje się i uruchamia w konsoli, co udokumentowano zrzutem ekranu

R.2.9	W programie tworzone są dwa obiekty klasy notatka, pierwszy ma identyfikator równy 1, drugi - 2 . Wartość licznika jest zgodna ze stanem rzeczywistym i jest równa 1, gdy jest utworzony tylko jeden obiekt, 2 gdy dwa obiekty itd. (sprawdzić w kodzie jaka jest kolejność tworzenia obiektu względem wyświetlania danych diagnostycznych)
R.3	**Rezultat 3: Aplikacja mobilna**
	Uwaga: jeżeli jest to możliwe uruchomić aplikację na tym samym urządzeniu, na którym uruchamiał zdający. Należy uwzględnić różnice pomiędzy emulacjami - takie cechy jak marginesy, wielkości bloków itp. nie należy brać pod uwagę. Kryteria 3.1 ÷ 3.7 sprawdzić w kodzie źródłowym, sprawdzane elementy muszą być zapisane zgodnie ze składnią. *Gdy aplikacja nie uruchamia się, a zdający zapisał zrzuty ekranu z uruchomienia aplikacji należy sprawdzić powód braku kompilacji. Jeśli występują błędy w plikach źródłowych zdającego kryteria 3.8 ÷ 3.10 nie są spełnione. Jeżeli błędy występują w innych plikach lub bibliotekach sprawdzić w kodzie oraz na zrzucie ekranu*
R.3.1	Zastosowano język znaczników XML/XAML lub inny do opisu interfejsu użytkownika oraz kod zawiera przynajmniej jeden element / kontrolkę interfejsu graficznego
R.3.2	Zastosowano rozkład liniowy wertykalny (LinearLayout / StackLayout lub inny o tej idei) z zagłębionym rozkładem liniowym horyzontalnym dla pola edycyjnego i przycisku
R.3.3	Zastosowano kontrolki: edycyjną, przycisku o treści DODAJ oraz widoku listy (np. ListView)
R.3.4	Nadano kolory dla przycisku: tło Crimson (#DC143C), czcionka biała
R.3.5	Dla widoku listy ustawiono kolor separatora: Crimson
R.3.6	Zdefiniowano zmienną dowolnej kolekcji o typie napisowym do przechowywania notatek, np. String[], ArrayList<String>, ObservableCollection<String> lub inne
R.3.7	Zdefiniowano funkcję powiązaną ze zdarzeniem kliknięcia przycisku. Funkcja dodaje do kolekcji treść wpisaną w pole edycyjne
R.3.8	W stanie początkowym aplikacja wyświetla trzy notatki o treści zgodnej z plikiem *dane.txt*
R.3.9	Po wpisaniu treści do pola edycyjnego i wybraniu przycisku jest ona wyświetlana jako ostatni element widoku listy
R.3.10	Aplikacja kompiluje się i uruchamia w emulatorze, co udokumentowano zrzutem ekranu jej układ jest zgodny z obrazem 1a lub 1b w arkuszu egzaminacyjnym. Pole edycyjne wyświetla podpowiedź "Nowy element". Separator jest widoczny (np. w Android Studio ustawiono wysokość separatora)
R.4	**Rezultat 4: Dokumentacja aplikacji konsolowej**
	Uwaga: nagłówek z kryteriów 4.1 ÷ 4.5 musi być zgodny ze stanem faktycznym z kodu źródłowego, nawet jeżeli w kodzie są błędy logiczne (liczba pól, typy). *Zrzuty ekranu z kryteriów 4.6 i 4.7 powinny zawierać cały obszar ekranu z widocznym paskiem zadań.* *Dokumentacja z kryterium 4.8 zapisana jest w pliku egzamin*
R.4.1	Dla klasy z aplikacji konsolowej zapisano nagłówek w postaci komentarza zgodny z Listingiem 1 z arkusza egzaminacyjnego (nie liczymy gwiazdek), komentarz może być wieloliniowy lub kilka jednoliniowych
R.4.2	W komentarzu ujęto nazwę i opis działania klasy
R.4.3	W komentarzu ujęto nazwy wszystkich pól klasy
R.4.4	Dla pól, które ujęto w komentarzu zapisano opis
R.4.5	W komentarzu ujęto numer zdającego
R.4.6	Zapisano przynajmniej jeden zrzut ekranu z uruchomienia lub kompilacji aplikacji konsolowej, na zrzucie widoczne jest środowisko, w którym powstała aplikacja
R.4.7	Zapisano przynajmniej jeden zrzut ekranu z uruchomienia lub kompilacji aplikacji mobilnej, na zrzucie widoczne jest środowisko, w którym powstała aplikacja
R.4.8	Dokumentacja zawiera: nazwę systemu operacyjnego, nazwy środowisk, emulatora, nazwy języków programowania