Nazwa kwalifikacji:	**Projektowanie, programowanie i testowanie aplikacji**
Oznaczenie kwalifikacji:	**INF.04**
Numer zadania:	**02**
Kod arkusza:	**INF.04-02-25.06-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, kryteria zastosować do aplikacji desktopowej. Kryteria dotyczą wyłącznie samodzielnie napisanego kodu. *Wystarczy, że sprawdzaną cechę zastosowano dla większości przypadków w kodzie (90%)*
R.1.1	Kod źródłowy zapisany w sposób czytelny: instrukcje w osobnych liniach, stosowane spacje pomiędzy operatorami, konsekwentnie stosowana wybrana konwencja dla nawiasów klamrowych instrukcji blokowej
R.1.2	Kod zapisany z wcięciami dla zagnieżdżeń bloków
R.1.3	Użyte polskie lub angielskie nazewnictwo metod / funkcji. Nazewnictwo jest znaczące
R.1.4	Użyte polskie lub angielskie nazewnictwo pól lub zmiennych. Nazewnictwo jest znaczące. Wyjątkami od reguły są zmienne bufor, tmp, iteratory pętli. Kryterium **nie jest** spełnione tylko wtedy, gdy zmienne nie istnieją lub ich nazwy nic nie znaczą, np. x, tab, tablica, foo, kolekcja
R.1.5	Zastosowane typy pasują do problemu, np. int, string
R.1.6	Podjęta próba uruchomienia kodu, co jest udokumentowane zrzutem przedstawiającym uruchomiony program lub jego kompilację
R.1.7	Program podejmuje zrozumiałą komunikację z użytkownikiem: komunikaty są znaczące, np. "Podaj tekst", "Podaj klucz"
R.2	**Rezultat 2: Aplikacja konsolowa**
	Uwaga: kryteria 2.1 ÷ 2.7 należy sprawdzić w pliku z kodem źródłowym 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 przypadku języka Python argument self nie jest wliczany do liczby parametrów oraz deklaracja pól w konstruktorze przez self. np. self.pole*
R.2.1	Algorytm *(nie musi być poprawny)* jest zaimplementowany w funkcji lub metodzie zwracającej wartość typu napisowego
R.2.2	Metoda/funkcja przyjmuje dwa parametry wejściowe typu napisowego i całkowitego, wartości tych parametrów zostały wczytane z klawiatury. Dopuszcza się w podejściu obiektowym metodę bezparametrową, jeśli tekst jawny i klucz są polami klasy
R.2.3	Metoda/funkcja zawiera pętlę działającą na wszystkich literach tekstu jawnego *(wejściowego)*
R.2.4	Gdy klucz = 0, tekst nie jest zmieniany. Znak spacji nie jest szyfrowany
R.2.5	Algorytm dla każdej litery tekstu wejściowego przesuwa ją o wartość klucza
R.2.6	Algorytm przy przesunięciu poza ostatnią/pierwszą literę alfabetu powoduje zawijanie *(powrót do jego początku/końca)* . Metoda dowolna, np. warunki, modulo, lub inne
R.2.7	Szyfr działa poprawnie dla klucza większego od długości alfabetu *(np. klucz = klucz % 26 lub inna metoda*)
R.2.8	Program uruchamia się w konsoli, co jest udokumentowane zrzutem ekranu
R.2.9	Po wprowadzeniu tekstu jawnego i wartości klucza w konsoli jest wyświetlany zaszyfrowany tekst *(na zrzucie i obowiązkowo w kodzie)*
R.3	**Rezultat 3: Aplikacja desktopowa**

	Uwaga: Kryteria 3.1 ÷ 3.6 sprawdzić w pliku z kodem źródłowym 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 3.7 ÷ 3.10 nie są spełnione. Jeżeli błędy występują w innych plikach lub bibliotekach, sprawdzić w kodzie oraz na zrzutach ekranu.* *Dopuszcza się literówki w wyświetlanych napisach*
R.3.1	Utworzona aplikacja desktopowa z tytułem okna "Szyfrowanie. Wykonane przez" dalej wstawiony numer zdającego, aplikacja zawiera co najmniej jedną kontrolkę wynikającą z treści zadania
R.3.2	W oknie zawarte: trzy napisy, pole edycyjne i wielowierszowe, pole do wypisywania szyfru i dwa przyciski
R.3.3	Zastosowane kolory tła: CadetBlue *(#5F9EA0)* dla okna, LightBlue *(#ADD8E6)* dla przycisków
R.3.4	Zastosowane kolory czcionki: AntiqueWhite *(#FAEBD7)* napisów *(etykiet)* , AliceBlue *(#F0F8FF)* dla pola z tekstem zaszyfrowanym
R.3.5	Zdefiniowana przynajmniej jedna metoda obsługująca kliknięcie przycisku *(może być pusta)*
R.3.6	Logika aplikacji jest obsłużona przy wykorzystaniu metody / funkcji szyfrującej
R.3.7	W stanie początkowym aplikacja wyświetla wszystkie kontrolki rozmieszczone zgodnie z obrazem 1 arkusza egzaminacyjnego *(w uruchomionej aplikacji lub na zrzucie i obowiązkowo w kodzie)*
R.3.8	Po wprowadzeniu tekstu i klucza oraz wybraniu przycisku „Zaszyfruj” zostaje wypisany tekst zaszyfrowany *(w uruchomionej aplikacji lub na zrzucie i obowiązkowo w kodzie)*
R.3.9	Po wybraniu przycisku "Zapisz w pliku" pojawia się okno zapisywania do pliku *(w uruchomionej aplikacji lub na zrzucie i obowiązkowo w kodzie)*
R.3.10	W zapisanym pliku znajduje się tekst zwrócony funkcją szyfrującą niezależnie od jej poprawności
R.4	**Rezultat 4: Testy aplikacji**
	Uwaga: kryteria 4.1, 4.3 ÷ 4.7 należy sprawdzić w kodzie. Wyniki testów muszą odzwierciedlać rzeczywisty stan - jeżeli funkcja źle działa dla danego testu, jest on zakończony niepowodzeniem. W kodzie należy sprawdzić, czy test rzeczywiście odwołuje się do funkcji szyfrującej i czy zawiera poprawne dane wejściowe i porównanie wyniku zwróconego z oczekiwanym *Zrzuty ekranu z kryteriów 4.8 i 4.9 muszą zawierać cały obszar ekranu z widocznym paskiem zadań. Dokumentacja z kryterium 4.10 zapisana jest w pliku egzamin*
R.4.1	Zapisana co najmniej jedna metoda testująca, w której została wykorzystana asercja, zastosowane nazwy odzwierciedlające cel testu
R.4.2	Co najmniej 1 metoda testująca porównuje wartość oczekiwaną z wartością zwracaną przez metodę szyfrującą i jest ona uruchomiona, co jest udokumentowane zrzutem ekranowym
R.4.3	Zapisany test jednostkowy 1, który przy danych wejściowych abc, 3 oczekuje wyniku def
R.4.4	Zapisany test jednostkowy 2, który przy danych wejściowych xyz, 3 oczekuje wyniku abc
R.4.5	Zapisany test jednostkowy 3, który przy danych wejściowych def, -3 oczekuje wyniku abc
R.4.6	Zapisany test jednostkowy 4, który przy danych wejściowych abc, 29 oczekuje wyniku def
R.4.7	Zapisany test jednostkowy 5, który przy danych wejściowych ab cd, 2 oczekuje wyniku cd ef
R.4.8	Zapisany przynajmniej jeden zrzut ekranu z uruchomienia lub kompilacji aplikacji konsolowej, na zrzucie widoczne jest środowisko, w którym powstała aplikacja
R.4.9	Zapisany przynajmniej jeden zrzut ekranu z uruchomienia lub kompilacji aplikacji desktopowej, na zrzucie widoczne jest środowisko, w którym powstała aplikacja

R.4.10	Dokumentacja zawiera: nazwę systemu operacyjnego, nazwy środowisk programistycznych, nazwy języków programowania *(stosowanych przez zdającego)* . Na płycie wszystkie wykonane projekty są spakowane oraz umieszczone pliki źródłowe i pliki wykonywalne (jeśli istnieją) na zewnątrz archiwów