

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

Oznaczenie kwalifikacji: **INF.04**

Numer zadania: **02**

Wersja arkusza: **SG**

Wypełnia zdający

Numer PESEL zdającego*

--	--	--	--	--	--	--	--	--	--	--	--	--

Miejsce na naklejkę z numerem
PESEL i z kodem ośrodka

Czas trwania egzaminu: **180** minut.

INF.04-02-22.06-SG

EGZAMIN ZAWODOWY

Rok 2022

CZĘŚĆ PRAKTYCZNA

**PODSTAWA PROGRAMOWA
2019**

Instrukcja dla zdającego

1. Na pierwszej stronie arkusza egzaminacyjnego wpisz w oznaczonym miejscu swój numer PESEL i naklej naklejkę z numerem PESEL i z kodem ośrodka.
2. Na KARCIE OCENY w oznaczonym miejscu przyklej naklejkę z numerem PESEL oraz wpisz:
 - swój numer PESEL*,
 - oznaczenie kwalifikacji,
 - numer zadania,
 - numer stanowiska.
3. Sprawdź, czy arkusz egzaminacyjny zawiera 6 stron i nie zawiera błędów. Ewentualny brak stron lub inne usterki zgłoś przez podniesienie ręki przewodniczącemu zespołu nadzorującego.
4. Zapoznaj się z treścią zadania oraz stanowiskiem egzaminacyjnym. Masz na to 10 minut. Czas ten nie jest wliczany do czasu trwania egzaminu.
5. Czas rozpoczęcia i zakończenia pracy zapisze w widocznym miejscu przewodniczący zespołu nadzorującego.
6. Wykonaj samodzielnie zadanie egzaminacyjne. Przestrzegaj zasad bezpieczeństwa i organizacji pracy.
7. Po zakończeniu wykonania zadania pozostaw arkusz egzaminacyjny z rezultatami oraz KARTĘ OCENY na swoim stanowisku lub w miejscu wskazanym przez przewodniczącego zespołu nadzorującego.
8. Po uzyskaniu zgody zespołu nadzorującego możesz opuścić salę/miejsce przeprowadzania egzaminu.

Powodzenia!

* w przypadku braku numeru PESEL – seria i numer paszportu lub innego dokumentu potwierdzającego tożsamość

Zadanie egzaminacyjne

UWAGA: katalog z rezultatami pracy oraz płytę należy opisać numerem, którym został podpisany arkusz, czyli numerem PESEL lub w przypadku jego braku numerem paszportu.

Wykonaj aplikację konsolową oraz webową według wskazań. Wykonaj dokumentację do aplikacji konsolowej, zgodnie z opisem w części III instrukcji do zadania. W tym celu zaloguj się na konto **Egzamin** bez hasła.

Utwórz folder i nazwij go swoim numerem. W folderze utwórz podfoldery: *konsoła*, *web*, *testy*. Po wykonaniu każdej aplikacji, jej pełny kod (cały folder projektu) **spakuj do archiwum**. Następnie pozostaw w podfolderze jedynie pliki źródłowe, których treść modyfikowałeś, plik uruchomieniowy, jeśli jest to możliwe oraz spakowane archiwum. Zrzuty ekranu dokumentujące uruchomienie obu aplikacji umieść w folderze *testy*.

Część I. Aplikacja konsolowa

Korzystając z mechanizmów programowania obiektowego zaprojektuj część logiki systemu forum użytkowników. Zaimplementuj klasę o nazwie *osoba* z konstruktorami, metodą i obsługą pola statycznego.

Założenia klasy:

- Zastosowano obiektowy język programowania zgodny z zainstalowanym na stanowisku egzaminacyjnym: C++ lub C#, lub Java, lub Python.
- Klasa *Osoba* zawiera:
 - Dwa pola: numeryczne id oraz tekstowe imię osoby. Dostęp do obu pól ma jedynie klasa. W przypadku późniejszego rozszerzenia klasy, klasy potomne **nie mają** dostępu do tych pól.
 - Ogólnie dostępne pole statyczne zliczające liczbę instancji klasy, początkowo wypełnione wartością 0.
 - Trzy konstruktory:
 - bezparametrowy, ustawia wartości 0 i pusty dla pól,
 - z dwoma parametrami, które przekazują wartości id i imienia,
 - kopiujący.
 - Każdy z konstruktorów dodatkowo inkrementuje liczbę instancji klasy.
 - Metodę do wypisania imienia obiektu klasy *Osoba* w postaci: „Cześć <argument>, mam na imię <imie>”, gdzie pole <argument> jest innym imieniem przekazanym jako parametr wejściowy metody. Jeżeli w obiekcie nie wypełniono wartości pola z imieniem wyświetlany jest komunikat „Brak danych”.

Uwaga: W języku Python, zamiast przeciążenia konstruktorów, należy utworzyć konstruktor z dwoma parametrami o wartościach domyślnych (0, pusty napis), który będzie zastępował konstruktor bezparametrowy oraz konstruktor z dwoma parametrami. Następnie zaimplementować metodę kopiującą dane z jednego obiektu do drugiego.

- Program powinien być zapisany czytelnie, z zachowaniem zasad czystego formatowania kodu, należy stosować znaczące nazwy pól i metod.
- Do klasy należy dołączyć dokumentację oraz testy w postaci programu głównego. Testy zostały opisane w części III zadania egzaminacyjnego.

Kod aplikacji przygotuj do nagrania na płytę. W podfolderze *konsoła* powinno znaleźć się archiwum całego projektu o nazwie *konsoła.zip*, plik z kodem źródłowym klasy oraz plik uruchomieniowy, jeżeli istnieje.

Część II. Aplikacja Web

Wykonaj aplikację internetową typu front-end obsługującą zapisy na kursy z zastosowaniem dostępnego na stanowisku egzaminacyjnym frameworka Angular lub biblioteki React. Zastosuj bibliotekę Bootstrap do zdefiniowania stylu formularza.

Liczba kursów: 3

1. Programowanie w C#
2. Angular dla początkujących
3. Kurs Django

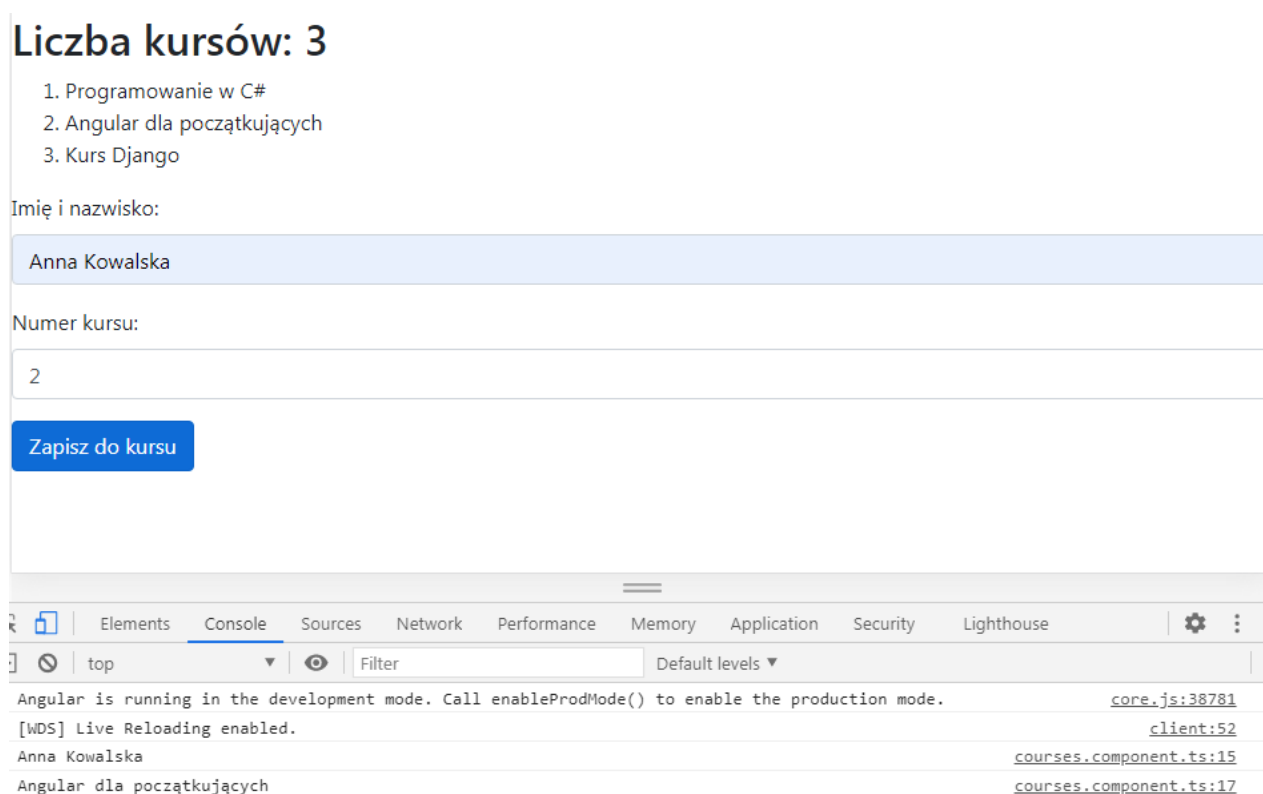
Imię i nazwisko:

Anna Kowalska

Numer kursu:

2

Zapisz do kursu



```
Angular is running in the development mode. Call enableProdMode() to enable the production mode. core.js:38781
[WDS] Live Reloading enabled. client:52
Anna Kowalska courses.component.ts:15
Angular dla początkujących courses.component.ts:17
```

Obraz 1a. Aplikacja framework Angular.

Liczba kursów: 3

1. Programowanie w C#
2. Angular dla początkujących
3. Kurs Django

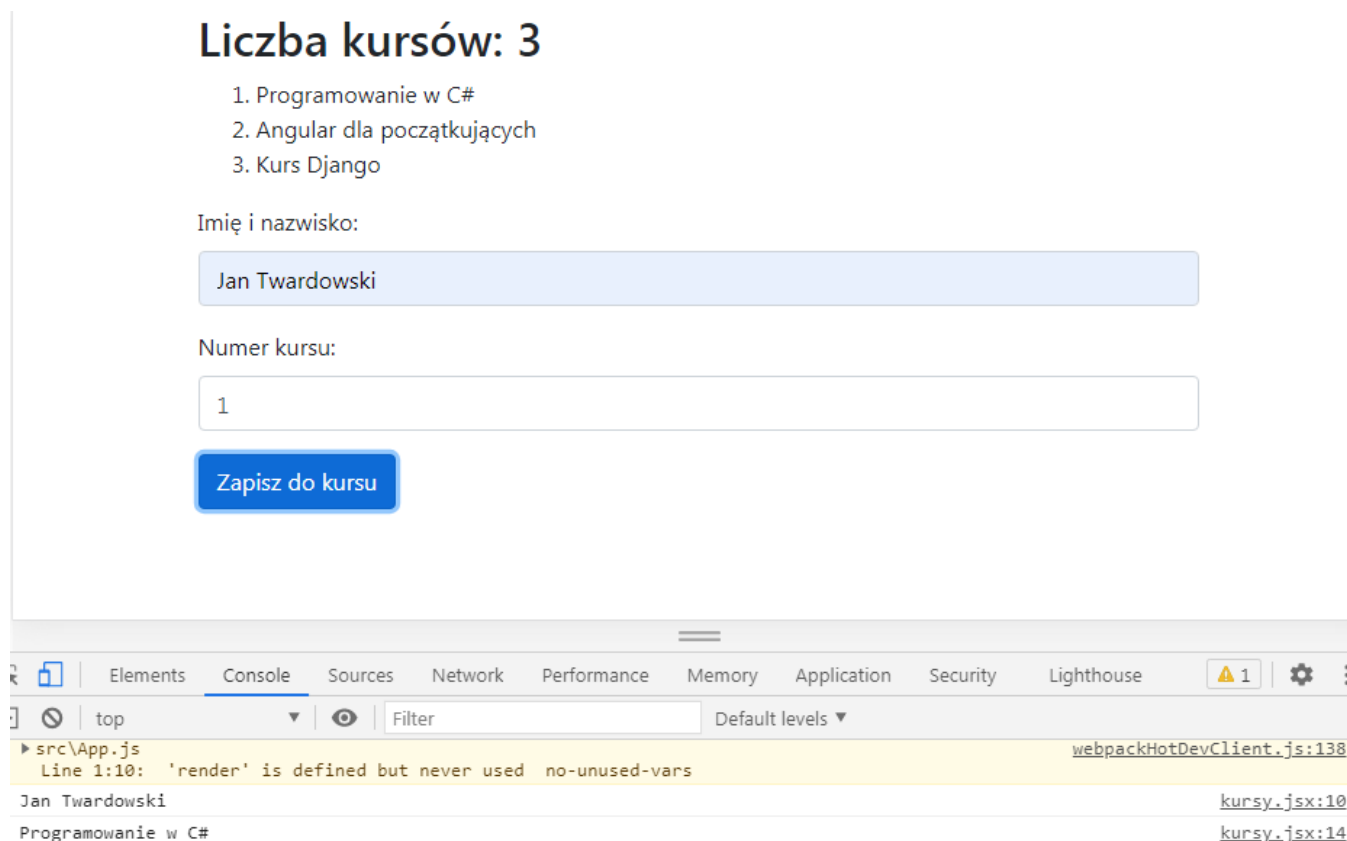
Imię i nazwisko:

Jan Twardowski

Numer kursu:

1

Zapisz do kursu



```
src\App.js webpackHotDevClient.js:138
Line 1:10: 'render' is defined but never used no-unused-vars
Jan Twardowski kursy.jsx:10
Programowanie w C# kursy.jsx:14
```

Obraz 1b. Aplikacja React.js.

Na obrazach 1a i 1b przedstawiono działanie aplikacji przygotowanej w środowisku Angular i React.js, stan po wybraniu przycisku „Zapisz do kursu”. W konsoli widoczne wyświetlenie imienia i nazwiska oraz nazwy kursu na podstawie danych wprowadzonych do formularza

Założenia aplikacji:

- Aplikacja składa się z jednego komponentu.
- Danymi komponentu jest tablica z nazwami kursów, o elementach: "Programowanie w C#", "Angular dla początkujących", "Kurs Django". Dla uproszczenia tablica może być zdefiniowana jako pole komponentu. Należy założyć, że tablica w przyszłości może się zmienić, co będzie miało wpływ na zachowanie i wygląd aplikacji.
- Komponent wyświetla:
 - Nagłówek drugiego stopnia o treści: „Liczba kursów: <liczba>”, gdzie <liczba> oznacza wielkość tablicy z nazwami kursów
 - Listę numerowaną generowaną automatycznie dla wszystkich elementów tablicy, niezależnie od jej wymiaru
 - Formularz składający się z:
 - pola edycyjnego i jego etykiety o treści „Imię i nazwisko:”
 - edycyjnego pola numerycznego i jego etykiety o treści „Numer kursu:”
 - przycisku „Zapisz do kursu”
- Aplikacja w stanie początkowym wyświetla puste pola formularza
- Elementy formularza są formatowane zgodnie z obrazem 1a lub 1b za pomocą stylów biblioteki Bootstrap. Do budowy szablonu HTML należy wykorzystać pomoc zamieszczoną w Tabeli 1. Należy zastosować znaczące nazwy dla identyfikatorów pól formularza
- Po wybraniu przycisku formularza jest generowane zdarzenie zatwierdzenia formularza, które wyświetla w konsoli przeglądarki:
 - Wartość wpisaną w pierwszym polu formularza
 - Nazwę kursu odpowiadającą numerowi wpisanemu w drugie pole formularza, gdy kurs o takim numerze istnieje. W przeciwnym wypadku wyświetla komunikat "Nieprawidłowy numer kursu"
- Aplikacja powinna być zapisana czytelnie, z zachowaniem zasad czystego formatowania kodu, należy stosować znaczące nazwy zmiennych i funkcji
- Dokumentacja do programu wykonana zgodnie z wytycznymi z części III zadania egzaminacyjnego.

Kod aplikacji przygotuj do nagrania na płytę. W podfolderze *web* powinno znaleźć się archiwum całego projektu o nazwie *web.zip* oraz pliki z kodem źródłowym, które były modyfikowane.

Część III. Testy utworzonych aplikacji

Wykonaj testy aplikacji konsolowej oraz dokumentację do aplikacji utworzonych na egzaminie.

W podfolderze *konsola* w programie głównym aplikacji konsolowej należy sprawdzić działanie klasy poprzez, kolejno:

- Wyświetlenie komunikatu „Liczba zarejestrowanych osób to <licznik>”, gdzie <licznik> jest wartością pobraną z pola statycznego klasy.
- Utworzenie obiektu za pomocą konstruktora bezparametrowego.
- Utworzenie obiektu za pomocą konstruktora z dwoma parametrami. Dane obiektu wprowadzane z klawiatury.
- Utworzenie obiektu za pomocą konstruktora kopiującego (w Python konstruktora bezparametrowego i metody kopiującej). Obiekt z wypełnionymi polami jest źródłem kopiowania danych.
- Wywołanie metody do wypisania imienia z parametrem wejściowym równym „Jan” dla wszystkich utworzonych obiektów.
- Ponowne wyświetlenie komunikatu „Liczba zarejestrowanych osób to <licznik>”, gdzie <licznik> jest wartością pobraną z pola statycznego klasy

Wykonaj zrzuty ekranu dokumentujące uruchomienie aplikacji utworzonych podczas egzaminu. Zrzuty powinny obejmować cały obszar ekranu monitora z widocznym paskiem zadań. Jeżeli aplikacja uruchamia się, na zrzucie należy umieścić okno z wynikiem działania programu oraz otwarte środowisko programistyczne z projektem lub okno terminala z kompilacją projektu. Jeżeli aplikacja nie uruchamia się z powodu błędów kompilacji, należy na zrzucie umieścić okno ze spisem błędów i widocznym otwartym środowiskiem programistycznym. Wykonać należy tyle zrzutów ile interakcji podejmuje aplikacja. Wymagane zrzuty ekranu:

- Aplikacja konsolowa – dowolna liczba zrzutów nazwanych *konsola1*, *konsola2* ...
- Aplikacja web – dowolna liczba zrzutów nazwanych *web1*, *web2* ... (np. stan początkowy, po wypełnieniu pól, po zatwierdzeniu konsola przeglądarki, gdy w drugim polu formularza jest prawidłowy numer oraz gdy jest nieprawidłowy numer kursu)

W edytorze tekstu pakietu biurowego utwórz plik z dokumentacją i nazwij go *egzamin*. Dokument powinien zawierać informacje:

- Nazwę systemu operacyjnego, na którym pracował zdający,
- Nazwy środowisk programistycznych, z których zdający korzystał na egzaminie,
- Nazwy języków programowania / frameworków / bibliotek użytych podczas tworzenia aplikacji,

Zrzuty ekranu i dokument umieść w folderze o nazwie *testy*.

UWAGA: Nagraj płytę z rezultatami pracy. W folderze z numerem zdającego powinny się znajdować podfoldery: *konsola*, *testy*, *web*. W folderze *konsola*: spakowany cały projekt aplikacji konsolowej, pliki z kodem źródłowym, opcjonalnie plik uruchomieniowy. W folderze *testy*: pliki ze zrzutami oraz plik *egzamin*. W folderze *web*: spakowany cały projekt aplikacji web, pliki modyfikowane przez zdającego. Po nagraniu płyty sprawdź poprawność nagrania. Opisz płytę swoim numerem i pozostaw na stanowisku, zapakowaną w pudełku wraz z arkuszem egzaminacyjnym.

Czas przeznaczony na wykonanie zadania wynosi 180 minut.

Ocenie będą podlegać 4 rezultaty

- implementacja, kompilacja, uruchomienie programu,
- aplikacja konsolowa,
- aplikacja web,
- testy aplikacji.

Tabela 1. Wybrane elementy frameworka Angular, biblioteki React.js i biblioteki Bootstrap - przykłady

<p>Angular To use ngModel i ngForm add: import { FormsModule } from '@angular/forms'; in app.module.ts file. Add FormsModule to imports table</p> <p>To use Bootstrap add to styles.css: @import "~bootstrap/dist/css/bootstrap.css";</p>
<p>React.js To use Bootstrap add: import 'bootstrap/dist/css/bootstrap.css';</p>
<p>Bootstrap Forms (Źródło https://getbootstrap.com/docs/4.0/components/forms/) Be sure to use an appropriate type attribute on all inputs (e.g., email for email address or number for numerical information) to take advantage of newer input controls like email verification, number selection, and more. Textual form controls—like <input>s, <select>s, and <textarea>s—are styled with the .form-control class. Included are styles for general appearance, focus state, sizing, and more.</p> <pre> <form> <div class="form-group"> <label for="exampleInputEmail1">Email address</label> <input type="email" class="form-control" id="exampleInputEmail1" /> </div> <div class="form-group"> ... </div> </form> </pre> <p>Important! In React render method use className instead of class; htmlFor instead of for.</p>
<p>Bootstrap buttons (Źródło https://getbootstrap.com/docs/4.0/components/buttons/) Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control. The btn classes are designed to be used with the <button> element. Add modifier classes as: btn-primary, btn-secondary, btn-success and more to btn class in order to add background colors. e.g. <button type="button" class="btn btn-success">Success</button></p>

Wypełnia zdający

**Do arkusza egzaminacyjnego dołączam płytę CD opisaną numerem PESEL

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

, której jakość nagrania została przeze mnie sprawdzona.**

Wypełnia Przewodniczący ZN

Potwierdzam, że do arkusza egzaminacyjnego dołączona jest płyta CD, opisana numerem PESEL zdającego.

.....
Czytelny podpis Przewodniczącego ZN