# SWA-Lab1. Zadania

Zajęcia są wprowadzające, więc im szybciej przedstawicie zrealizowane zadania, tym szybciej skończycie zajęcia. Zadania nie są porażająco trudne, ale zobaczymy w praniu.

1. Zaloguj się na swoim profilu studenckim.
2. Sprawdź możliwość zapisu (uprawnienia) na [gdrive](https://docs.google.com/document/d/1cgHTzsIgMghaFrujpxFmEK-ZcnMEuIZri7W2xAugKVE/edit?tab=t.0) inbox. Będziesz tam umieszczać raporty z realizacji zadań. Wymagane jest logowanie z konta studenckiego. Awaryjnie podczas zajęć jest dostępny [folder zrzutowy dla wszystkich](https://drive.google.com/drive/folders/11nGeJy8_uZhsMM3oshdLnlRtCKBB9wU1) - ale będą dodatkowe opłaty za korzystanie z niego.
3. Na początku zajęć pojawi się krótki Quiz (gdoc), rozwiąż go i w formie np. PDF lub TXT prześlij do inbox, tam gdzie raporty z realizacji zadań.
4. ~~Uruchom Visual Studio Code - ale nie instaluj dzisiaj rozszerzenia Pico 🙁 Opiekun sali pracuje nad tym, aby ten dodatek był zainstalowany globalnie w systemie. Nawet jeśli Pico ext. jest zainstalowane, to podczas tworzenia projektu przy próbie pobrania SDK zabraknie miejsca na dysku (quota).~~
5. ~~Jeśli masz problemy z logowaniem lub uruchomieniem Visual Studio Code, to zgłoś ten problem prowadzącemu zajęcia.~~
6. ~~Spróbuj zainstalować program Thonny.~~

Jeśli masz przy sobie laptop, to kontynuuj zajęcia na nim - zestaw zadań blue pill🙂

Jeśli nie masz laptopa lub nie potrafisz uruchomić Code+Pico extension, to poproś prowadzącego o pomoc lub przejdź do zestawu zadań red pill.

## Zestaw blue pill

1. Utwórz projekt Lab1\_zad1, użyj języka C. Zadanie polega na wyświetleniu “Hello, world!” w terminalu szeregowym co jedną sekundę - jest to domyślny projekt, który zostanie wygenerowany przez kreator. Zmodyfikuj projekt tak, aby każdy wyświetlony napis był numerowany począwszy od 1, a po jego wyświetleniu LED szybko mignął 1..10 razy odpowiednio do wartości ostatniej cyfry licznika:   
   1 Hello, world! → LED miga 1 raz  
   …  
   99 Hello, world! → LED miga 9 razy  
   12300 Hello, world! → LED miga 10 razy  
   Szybko znaczy na tyle szybko, żeby 10 błysków zmieściło się przed następnym krokiem.
2. Utwórz projekt Lab1\_zad2, użyj języka C. Program ma reagować na znaki wprowadzone w terminalu szeregowym w taki sposób, aby małe litery łacińskie były zamieniane na wielkie i odsyłane z powrotem do terminala. Znaki ‘1’..’9’ mają rozpocząć odliczanie od N do 0 i co sekundę przypominać, że do detonacji zostało tyle właśnie sekund. W trakcie odliczania LED ma migać z częstotliwością 1 Hz, a gdy zostanie 3 sekundy lub mniej to z częstotliwością 10 Hz. W trakcie odliczania możesz ignorować przychodzące znaki, mamy wtedy znacznie większy problem.  
   a → A  
   B → B  
   ? → (brak)  
   5 → 5… 4… 3… 2… 1… kaboom!  
   **Wskazówka 1**: domyślnie stdin jest buforowane i blokujące, więc getchar() nie jest dobrym rozwiązaniem. zamiast tego możesz użyć getchar\_timeout\_us(...)  
   **Wskazówka 2**: Jeśli zaznaczysz komponent “UART” tworząc projekt, to pojawi się w nim możliwość używania UART bezpośrednio, a nie tylko poprzez stdio. Możesz wtedy użyć np. uart1 na pinach 4,5 zamiast uart0 na pinach 0,1 który domyślnie jest używany przez stdio. Co to daje? Łatwe jest sprawdzanie, czy jest jakiś znak do odebrania: uart\_is\_readable(...), odebranie go uart\_getc(...) oraz wysłanie znaku uart\_putc(...). Wysłanie napisu to uart\_puts(...), a formatowany napis można zbudować przy pomocy sprintf(...) zamiast printf(...).

## Zestaw red pill

1. Znajdź w sieci plik uf2 z najnowszą wersją MicroPython dla Pico2W. Pobierz go i zaprogramuj Pico tym wsadem → demonstracja na wykładzie lub na laborce.
2. Zainstaluj pakiet Thonny (mam nadzieję, że się da).
3. Wykonaj zadania z niebieskiej listy, ale używając języka MicroPython. Jeśli ChatGPT nie da rady, to poproś o pomoc prowadzącego zajęcia.

## Raport (sprawozdanie)

Do końca zajęć w swoim indywidualnym folderze zrzutowym “inbox” zgodnie z wytycznymi podanymi w instrukcji wrzuć pliki źródłowe i nagłówkowe (bez dodatkowych śmieci) oraz kilka zrzutów ekranu lub krótki filmik przedstawiający działanie twojego kodu.

## Zadanie domowe - na następne zajęcia, ale noc młoda ;)

Chodzi o oswojenie się z możliwościami MCU, nazewnictwem i lokalizacją peryferiów innych niż GPIO, oraz stopniowe zapoznawanie się z układami peryferyjnymi które będą dostępne na płycie głównej.

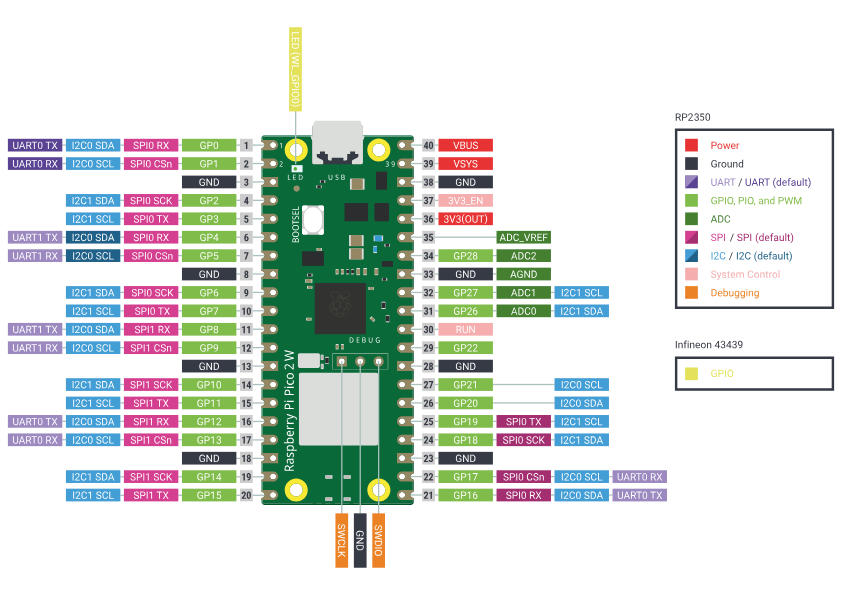
Możliwe jest też użycie innych czujników i układów wykonawczych, będziemy to na bieżąco uzgadniać gdyby pojawiły się ciekawe pomysły i propozycje z waszej strony.

Zapoznaj się z projektem <https://github.com/jmnich/beam_kit_1>. Plik PDF ze schematem [jest na gdrive](https://drive.google.com/drive/folders/1U_iFf3BdP5KhBDhqRGj7jW1A3Mkkr7G-).

Zapoznaj się z dokumentacją modułu Raspberry Pi Pico 2 W.

Zapoznaj się z dokumentacją mikrokontrolera RP2350.

(nie uczcie się na pamięć, tylko zapiszcie gdzieś pod ręką linki lub pobierzcie pliki, żeby mieć je na zajęciach łatwo dostępne i wiedzieć gdzie szukać takich informacji jak np. rozmieszczenie pinów). Najważniejszy schemat jest tutaj:



Model 3D zestawu dydaktycznego (produkcja PCB i montaż zakończone, aktualnie trwa pakowanie, czekamy na wysyłkę i odprawę celną):

