

Dział Software

Poniżej znajdują się treści dwóch zadań rekrutacyjnych do działu Software. Wykonaj jedno lub dwa z nich (w drugim zadaniu możesz wykonać dowolną ilość podpunktów – im więcej tym lepiej), a link do repozytorium odeślij na maila **wiktor.koczkodaj.pwrrt@gmail.com** do **17.10.2025r. do godziny 23.59**. W tytule maila wpisz [IMIE NAZWISKO] REKRUTACJA.

Rozwiązania obu zadań można zawrzeć w tym samym repozytorium.

Zadanie 1.

Twoim zadaniem jest:

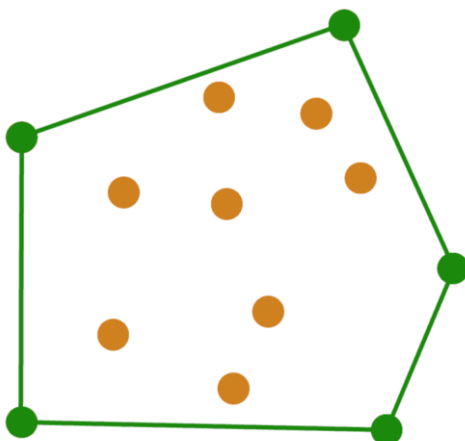
- Stworzenie projektu w programie STM32CubeIDE z konfiguracją do pracy z mikrokontrolerem STM32F205RGT6 pracującym z zewnętrznym źródłem taktowania o częstotliwości 16MHz
 - Napisanie programu, który zrealizuje funkcję mrugania diodą (włączania i wyłączania) z częstotliwością 5Hz. Dioda podłączona jest do pinu PA5.
- Do realizacji zadania wykorzystaj timer.
- Zaimplementowanie watchdoga do programu.

Kod powinien być udokumentowany, czyli wyjaśnione powinno być zarówno działanie funkcji, jak i ważniejsze fragmenty kodu. Ponadto do swojego rozwiązania załącz plik, w którym opiszesz swoje rozwiązanie. Kod i dokument umieść w repozytorium Git, do którego link wyślij jako rozwiązanie zadania.

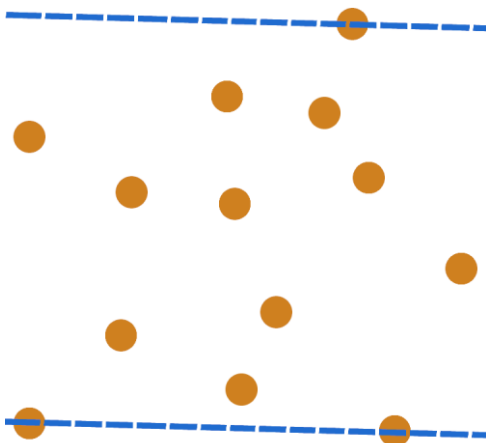
Zadanie 2.

Napisz w języku C++ program, który przyjmuje zbiór punktów na przestrzeni dwuwymiarowej i realizuje poniższe zadania:

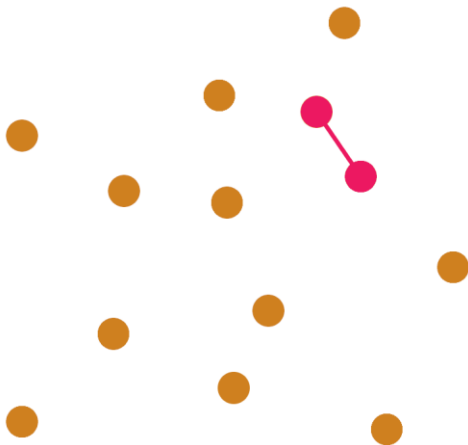
1. Znajduje najmniejszy (najmniej liczny) podzbiór takich punktów, które otaczają wszystkie inne punkty.



2. Znajduje najbliższe (najmniej oddalone od siebie nawzajem) dwie proste równoległe takie, że wszystkie punkty w zbiorze leżą dokładnie na nich lub między nimi.



3. Znajduje dwa najbliższe sobie położone punkty w zbiorze. Skup się na osiągnięciu jak najniższej złożoności obliczeniowej.



Program na wejściu powinien przyjmować nazwę pliku, w którym znajdować się będą punkty w następującym formacie:

- `<n>` - pierwsza linia pliku zawierać będzie liczbę punktów w zbiorze
- `<x> <y>` - w kolejnych `n` liniach znajdować się będą współrzędne tych punktów, współrzędna `x` oddzielona będzie spacją od współrzędnej `y`.

Współrzędne punktów nie muszą być liczbami całkowitymi. Punkty nie będą się pokrywać.

Program spełniający wszystkie podpunkty po skończeniu działania powinien wypisać w terminalu:

- Punkty tworzące otoczkę (kolejność w jakiej zostaną wypisane nie ma znaczenia).
- Odległość między znalezionymi prostymi równoległymi ograniczającymi wszystkie punkty (może być zaokrąglona).
- Dwa punkty w zbiorze leżące najbliżej siebie i odległość między nimi (również może być zaokrąglona, jeśli kilka par punktów dzieli ta sama minimalna odległość, program może wypisać dowolną z nich).

Przykładowe dane wyjściowe:

10

-4 2

2.5 1.5

4 3

0 -2

-2 4

4 -1.5

2 2

-2 1

-1 2.5

1 0

Przykładowe wyjście (w programie wykonującym wszystkie zadania):

Otoczka: $(-2.0, 4.0)$, $(-4.0, 2.0)$, $(0.0, -2.0)$, $(4.0, -1.5)$, $(4.0, 3.0)$

Proste: $d=5.5896$

Najbliższe Punkty: $[(2.0, 2.0), (2.5, 1.5)]$ $d=0.707$

Kod powinien być udokumentowany, czyli wyjaśnione powinno być zarówno działanie funkcji, jak i ważniejsze fragmenty kodu. Ponadto do swojego rozwiązania załącz plik, w którym opiszesz swoje rozwiązanie (sposób działania, zastosowane rozwiązania, algorytmy i technologie, napotkane edge-case'y bądź inne problemy i ich rozwiązania itp.). Kod i dokument umieść w repozytorium Git, do którego link wyślij jako rozwiązanie zadania. Znajomość i zrozumienie własnych rozwiązań zostanie zweryfikowana.

Powodzenia!