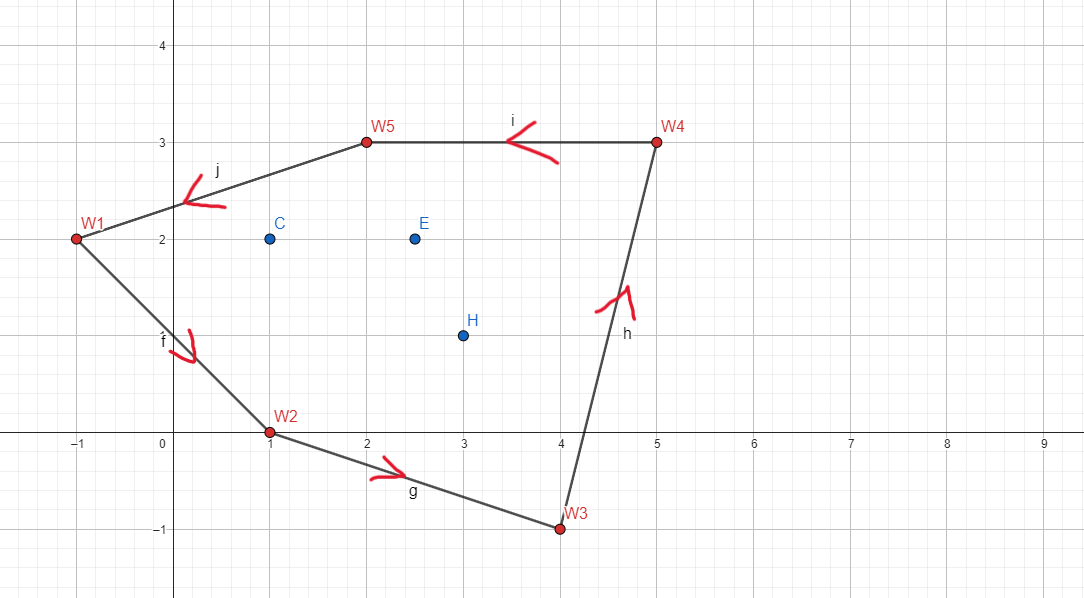
# Dokument opisujący sposób rozwiązania zadania rekrutacyjnego nr 2

Przede wszystkim mój program oparty jest na klasie points która to zawiera dwie zmienne typu double (aby możliwe było wczytywanie liczb również z dokładnością mniejszą niż 1). Bardzo ważnym elementem mojego programu są również struktury danych typu vector dzięki którym program ma pełną swobodę w manipulowaniu punktami w kontenerze zawierającym już znalezione wierzchołki jak i tym zawierającym pozostałe punkty.

Program rozpoczyna swoje właściwe działanie od znalezienia punktów wysuniętych najdalej w każdą ze stron obu osi. Ta część kodu wykonuje się w funkcji searching\_for\_the\_first\_vertices. W związku z tym, że nie zawsze występują, aż 4 punkty wysunięte w każdą ze stron(przykładowo jeden punkt może przyjmować wartość największą na osi X jak i Y), program akceptuje też sytuację gdy są tylko 3 takie punkty, a także dobiera sobie punkt gdy występują jedynie 2 tego rodzaju współrzędne. Wtedy również weryfikuje czy dobierany punkt znajduje się po lewej stronie pierwotnie utworzonej linii (z dwóch najbardziej wysuniętych punktów), aby zapewnić prawidłowe działanie programu. Jest to o tyle niezbędne, że program układa znalezione już wierzchołki w vecotrze apex przeciwnie do ruchu wskazówek zegara, więc dodanie punktu który odwraca kierunek przebiegu wierzchołków spowodowałoby błędy w działaniu programu.

Zobrazowałem to na poniższej grafice:

Kolejność ułożenia punktów w kontenerze -> {W1,W2,W3,W4,W5}

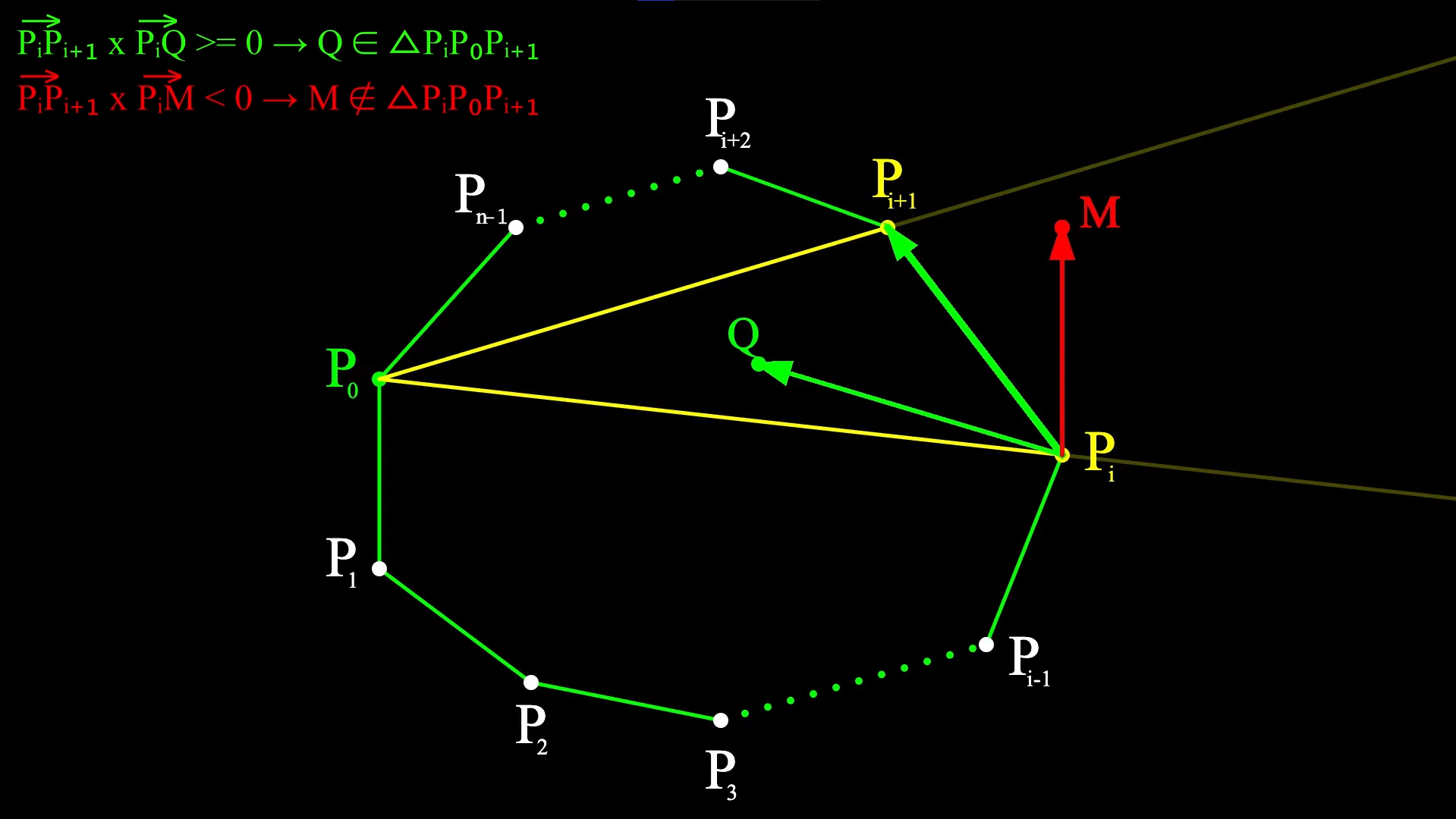


Za powyższe kroki weryfikujące odpowiedzialna jest funkcja validation.

Po tym program w funkcji organizing\_vectors sprawdza które punkty zostały oznaczone jako wierzchołki i usuwa je z vectora dostępnych punktów (vector points).

Następnie program przechodzi do poszukiwania i sprawdzania pozostałych punktów jako potencjalnych wierzchołków. Rolę tę spełnia funkcja searching\_another\_apaxes. Do znajdowania innych wierzchołków wykorzystałem algorytm "Point in Convex Polygon", który z pomocą iloczynu wektorowego weryfikuje czy sprawdzany punkt znajduje się wewnątrz już utworzonego wielokąta. Jeśli nie punkt oznaczany jest jako wierzchołek. Proces sprawdzania oblicza iloczyn wektorowy dwóch wektorów (przykładowo P1 i P2). P1 w tym przypadku jest wektorem łączącym już dwa odkryte wierzchołki (wierzchołek startowy oraz wierzchołek sprawdzany), a wektor P2 jest wektorem łączącym wierzchołek startowy z punktem sprawdzanym. Jeśli iloczyn wektorowy okaże się dodatni będzie to znaczyło że punkt leży wewnątrz otaczającego wielokąta. Swoją wiedzę na temat działania tego algorytmu czerpałem z [tego](https://www.youtube.com/watch?v=aoxOPx2BIHE) filmu.

Schemat z filmu opisującego zasadę działania algorytmu:



W programie na bieżąco, po każdorazowym dodaniu kolejnego wierzchołka do kontenera, działa funkcja weryfikująca, czy któryś z wierzchołków nie tworzy kąta wklęsłego. Jeśli taka sytuacja występuje, punkt ten usuwany jest z vecotra apex. To działanie zlecane jest przez funkcję removing\_invalid\_apexes. Proces ten odbywa się z pomocą iloczynu wektorowego oraz funkcji sinus, która sprawdza wartość sinusa dla danego kąta między wierzchołkami. Jeśli wartość ta jest ujemna, znaczy to, że kąt wykracza poza granice 180, a więc należy usunąć wierzchołek odpowiadający za niego. Funkcja ta zastosowana jest również pod samo koniec działania programu, aby ostatecznie zweryfikować poprawność utworzonego wielokąta, który otacza pozostałe punkty.