

# Zad. 4: Rotacje 2D

## 1 Cel ćwiczenia

Wykształcenie umiejętności modelowania kluczowych dla danego problemu pojęć. Definiowanie właściwego interfejsu klasy. Zwrócenie uwagi na dobór odpowiednich struktur danych w zależności od metody rozwiązywania problemu. Praktyczne zapoznanie się z problemem skończonej binarnej reprezentacji liczb oraz wynikającego stąd problemu niedokładności obliczeń.

## 2 Program zajęć

- *Demonstracja przykładu problemu skończonej reprezentacji binarnej liczb*
- *Ocena realizacji zadania z poprzedniego laboratorium* – ocenie podlega poprawność realizacji zadania, styl pisanie programu oraz opisy.
- *Ocena przygotowania do zajęć* – ocenie podlega diagram czynności (szczegóły wymagań patrz podrozdział 4)
- *Modyfikacja programu wg wskazań osoby prowadzącej* – ocenie będzie podlegała poprawność realizacji modyfikacji. Pracę nad modyfikacją programu (wszystkie operacje należy wykonywać na kopii) należy rozpocząć już w trakcie pierwszej fazy laboratorium, gdyż prowadzący nie będzie w stanie ocenić wcześniejszego programu wszystkim jednocześnie.
- *Realizacja wstępnej fazy prac nad nowym zadaniem* – w ramach wstępnej realizacji zadania należy zdefiniować klasę wektor oraz przeciążenia odpowiednich metod. Definicje należy wpisać w odpowiednim module dostarczonego załączka.
- *Ocena realizacji wstępnej fazy zadania*

## 3 Opis zadania programowego

Należy napisać program, który umożliwi rotację prostokąta o zadany kąt wokół środka układu współrzędnych. Po dokonanych obrocie współrzędne wierzchołków powinny zostać zaktualizowane, aby odzwierciedlały nowe położenie prostokąta. Program ma udostępniać proste menu, które pozwalać ma na następujące operacje:

- obrót prostokąta o zadany kąt z zadaną ilością powtórzeń operacji,
- przesunięcie prostokąta o zadany wektor,
- wyświetlenie współrzędnych wierzchołków,
- sprawdzenie długości przeciwległych boków,
- wyświetlenie menu,
- zakończenie działania programu.

Sprawdzenie długości przeciwległych boków ma na celu sprawdzenie, czy długość przeciwległych boków jest równa. Oprócz wspomnianych operacji program powinien wizualizować położenie prostokąta wykorzystując dostarczony moduł `LaczeDoGnuplot`.

## 4 Przygotowanie do zajęć

Należy przygotować diagram czynności dla operacji obrotu prostokąta o zadany kąt wokół środka układu współrzędnych.

## 5 Przykład działania programu

Poniżej przedstawiony przykład wyznacza formę komunikatów i ich forma jest obligatoryjny dla programu tworzego w ramach niniejszego zadania.

```
panamint> ./program_obroty_2D
```

```
o - obrot prostokata o zadany kat
p - przesuniecie prostokata o zadany wektor
w - wyswietlenie wspolrzecznych wierzchołkow
s - sprawdzenie dlugosci przeciwleglych bokow
m - wyswietl menu
k - koniec dzialania programu
```

```
Twoj wybor? (m - menu) > w
```

```
1.0000000000    1.0000000000
11.0000000000   1.0000000000
11.0000000000   6.0000000000
1.0000000000    6.0000000000
```

```
Twoj wybor? (m - menu) > s
```

```
:) Dłuższe przeciwległe boki są sobie równe.
:) Krótsze przeciwległe boki są sobie równe.
```

```
Twoj wybor? (m - menu) > p
```

```
Wprowadz wspolrzedne wektora translacji w postaci dwoch liczb
tzn. wspolrzednej x oraz wspolrzednej y.
```

```
1 1
```

```
Twoj wybor? (m - menu) > w
```

```
2.0000000000    2.0000000000
12.0000000000   2.0000000000
12.0000000000   7.0000000000
2.0000000000    7.0000000000
```

```
Twoj wybor? (m - menu) > o
```

```
Podaj wartosc kata obrotu w stopniach
```

```
1
```

```
Ile razy operacja obrotu ma byc powtorzona?
```

3600000

Twoj wybor? (m - menu) > w

```
2.1170041561    2.1025645733
12.6662225723   2.0672414303
12.6843624115   7.3417510986
2.1351695061    7.3789219856
```

Twoj wybor? (m - menu) > s

:0 Dłuższe przeciwległe boki nie są równe!!!

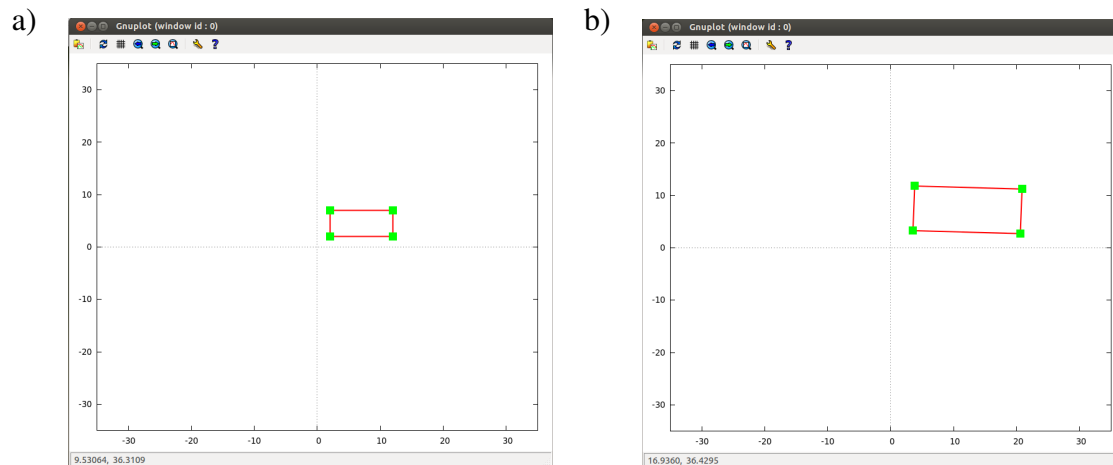
:0 Krótsze przeciwległe boki nie są równe!!!

Twoj wybor? (m - menu) > k

Koniec działania program

panamint> \_

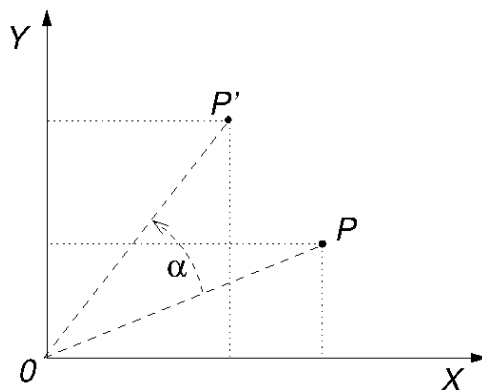
W przykładzie zademonstrowano kumulowanie się błędów obliczeń. Bardzo wyraźne i widoczne zniekształcenia pierwotnego prostokąta są widoczne przy zadaniu kąta rotacji  $1^\circ$  i powtórzenia tej operacji 36 000 000 razy. Porównanie obu prostokątów (pierwotnego i po zrealizowaniu ciągu transformacji) widoczny jest na rys. 1



Rysunek 1: Zestawienie prostokątów a) przed dokonaniem ciągu rotacji, b) po dokonaniu ciągu rotacji o  $1^\circ$  i powtórzenia jej 36 000 000 razy

## 6 Realizacja rotacji

Niech będzie dany punkt  $P = (x, y)$ . Dokonując rotacji tego punktu o kąt  $\alpha$  wokół środka układu współrzędnych otrzymujemy nowy punkt  $P' = (x', y')$  tak jak to jest pokazane na rysunku rys. 2. Transformację rotacji współrzędnych punktu  $P$  do współrzędnych punktu  $P'$  realizujemy



Rysunek 2: Rotacja punktu  $P$  o kąt  $\alpha$

zgodnie z następującym wzorem:

$$\begin{aligned}x' &= x \cos \alpha - y \sin \alpha, \\y' &= x \sin \alpha + y \cos \alpha.\end{aligned}$$

Możemy go zapisać w postaci macierzowej

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Tak więc mając na uwadze, że  $P = (x, y)$  i  $P' = (x', y')$  oraz oznaczając macierz rotacji  $\mathbf{R}_\alpha$ , powyższą wzór można zapisać w formie

$$P' = \mathbf{R}_\alpha \cdot P.$$

W programie należy dokonać odpowiednich przeciążeń operatorów, aby tego typu zapis można było stosować bezpośrednio w programie.

## 7 Materiały pomocnicze

Załączek programu znajduje się w katalogu `~bk/edu/kpo/zad/z4`. Zawiera on przykład wykorzystania modułu złącza do programu `gnuplot`, który pozwala zwizualizować wyniki obliczeń. Podstawowe objaśnienia znajdują się w kodzie dostarczonego załączka.

## 8 Wymagania co do konstrukcji programu

Oprócz wymagań sformułowanych w opisie zadania należy uwzględnić uwarunkowania przedstawione poniżej.

- Należy zdefiniować klasy `Wektor2D`, `Macierz2x2` oraz `Prostokat`. Muszą one mieć **tylko i wyłącznie niezbędne pola reprezentujące atrybuty danego pojęcia**.
- Należy odpowiednio przeciążyć operator indeksujący dla klasy `Wektor2D`, operator funkcyjny dla klasy `Macierz2x2` i odpowiednio posługiwać się nimi w programie. Możliwe są również również inne kombinacje tych operatorów.

- Należy przeciążyć operator mnożenia, tak aby była możliwość przemnożenia macierzy przez wektor. Ponadto konieczne jest przeciążenie operatora dodawania i odejmowania wektorów.
- Program musi zachować strukturę modułową i odpowiednią strukturę kartotek. O ile będzie to konieczne, należy zmodyfikować plik `Makefile` (np. gdy dodany zostanie nowy moduł).
- Każda z klas powinna zostać zdefiniowana w oddzielnym pliku nagłówkowym. Metody tej klasy powinny być natomiast definiowane w osobnym module związanym z daną klasą, np. definicja klasy `Prostokat` powinna znaleźć się w pliku nagłówkowym `Prostokat.hh`, zaś metody w pliku `Prostokat.cpp`. Proste metody można definiować bezpośrednio w ciele klasy.
- Dla poszczególnych klas należy przeciążyć niezbędne operatory działające na strumieniach. Nie wszystkie przeciążenia są w tym zadaniu potrzebne. Na pewno będą potrzebne przeciążenia operatorów wczytywania i zapisu dla klasy `Wektor2D` oraz operatora wyświetlania dla klasy `Prostokat`.
- Wszystkie metody, które nie zmieniają stanu obiektu, na którym działają, powinny być metodami typu `const`.
- Wszystkie klasy i metody oraz funkcje powinny zostać opisane zgodnie z zaleceniami przedstawionymi w pliku `~bk/edu/kpo/zalecenia.txt`.
- Dla końcowej wersji programu powinna zostać wygenerowana w formacie HTML za pomocą systemu `doxygen`.

Oprócz tego pozostają w mocy wszystkie wcześniejsze wymagania dotyczące struktury katalogów, pliku `Makefile`, modułowej struktury programu, jak też opisów.

## 9 Wymagania i zarys programu zajęć w okresie realizacji zadania

Przystępując do pracy nad programem należy pamiętać, że menu programu dodajemy na samym końcu, gdy stworzymy już i przetestujemy wszystkie niezbędne funkcjonalności. Zaczynanie pracy od menu nie jest dobrym rozwiązaniem.

### 9.1 Tydzień 0

Należy przygotować diagram czynności dla operacji obrotu prostokąta o zadany kąt wokół środka układu współrzędnych.

### 9.2 Tydzień 1

Przed zajęciami muszą zostać przygotowane następujące elementy zadania. Wszystko co będzie ponad to będzie oceniane *in plus* (oprócz menu programu). Zakłada się, że w programie będą przetestowane podstawowe funkcjonalności związane z obrotem i translacją prostokąta. W

tej wersji programu nie jest pożądane, aby występowało menu. Wyjątkiem jest sytuacja, gdy program zostanie wcześniej skończony.

- Zdefiniowane powinny być klasy `Wektor2D`, `Macierz2x2` oraz `Prostokat`. Wszystkie klasy muszą mieć zdefiniowane odpowiednie metody.
- Warunkiem koniecznym pozytywnej oceny jest poprawna kompilacja. W trakcie kompilacji nie powinny być generowane żadne ostrzeżenia.
- W funkcji `main` powinien być kod, który demonstruje rysowanie prostokąta, jego obrót o zadany kąt. Wartość tego kąta może być (a w celach testowych nawet należy aby była) wpisana na „sztywno” w kodzie programu. Po dokonaniu obrotu powinien pojawić się nowy rysunek. W analogiczny sposób powinna zostać zademonstrowana operacja translacji.

Uwaga: W tej wersji programu nie musi być menu.

- Powinny być wyświetlane współrzędne wierzchołków prostokąta. Muszą więc zostać przeciążone odpowiednie operatory dla klas `Wektor2D` i `Prostokat`, które umożliwią wypisanie współrzędnych na wyjście standardowe.
- Wszystkie klasy i metody muszą być opisane.

W trakcie zajęć zostanie zademonstrowana praca z systemem `doxygen`. Trzeba będzie odpowiednio zmodyfikować opisy, tak aby były rozumiane i właściwie przetworzone przez system `doxygen`.

## 9.3 Tydzień 2

Rozliczenie się z gotowego programu i rozpoczęcie następnego zadania.