Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Projekt języka graficznego 2D

Oskar Kocjan
Stanisław Światłoch
Szymon Auguścik

1 Opis języka

Celem naszego projektu jest stworzenie prostego języka graficznego 2D. Podstawową funkcjonalnością będzie tworzenie kształtów, takich jak proste, koła, kwadraty, wielokąty, oraz wykonywania na nich operacji zmiany barwy, przesuwania, skalowania i rotacji. Dodatkowo, zostaną zaimplementowane tablice, pętle, instrukcje warunkowe oraz podstawowa arytmetyka.

2 Składnia

2.1 Tworzenie obiektów

- punkt - point Name: X, Y
- koło - circle Name: CentralPoint, Radius
- odcinek - segment Name: BeginPoint, EndPoint
- wielokąt - polygon Name: GroupOfPoints

2.2 Podstawowe polecenia

2.2.1 Inicjalizacja

canvas: X, Y, Color
Color jest stałą, np. #red, #green, #transparent itd.

2.2.2 Rysowanie

draw Shape
Shape to dowolny kształt

2.3 Grupowanie

group Name: ...
... to dowolna liczba zmiennych (cn. 1)

2.4 Typy

Dostępne typy

1. point, circle, polygon, segment - rodzina podstawowych kształtów, które mogą zostać narysowane przez użytkownika
2. num - liczba zmiennoprzecinkowa
3. iterator - liczba całkowita dodatnia (używana w definicji pętli)
4. group - struktura przechowująca kilka wartości/obiektów

Dodatkowo w programie wyróżniony będzie typ logiczny potrzebny do utworzenia instrukcji warunkowej, jednakże użytkownik nie będzie mieć możliwości tworzenia zmiennych o danym typie. Będzie on wykorzystywany jedynie przy instrukcji check w postaci operacji logicznej.

2.5 Operatory

- +, -, *, / - podstawowe operatory arytmetyczne
- <=, >, >=, <, =, != - relacje logiczne
- ! - logiczna negacja
- % - modulo

Operatory arytmetyczne oraz modulo będą zdefiniowane jedynie dla dwóch typów danych: num oraz iterator. Pozostałe pozwolą na definiowanie odpowiednich operacji logicznych.

2.6 Rodzaje transformacji

- fill Shape: Color - wypełnienie zadanym kolorem
- move Shape: X, Y - przesunięcie o wektor [X; Y]
- place Shape: P - przesunięcie figury z pierwszego charakterystycznego punktu do punktu P
- rotate Shape: Angle, P - rotacja względem punktu P
- scale Shape: Rate - skalowanie

2.7 Instrukcja iteracyjna

loop Iter start Start until End step Step

    ...
end

- Iter - nazwa iterowanej zmiennej
- Start, End - wartości: początkowa i końcowa
- Step - krok iteracyjny
- ... - dowolna liczba operacji

2.8 Instrukcja warunkowa

check Cond1
    ...
else check Cond2
    ...
else
    ...
end

- Cond1, Cond2 - warunki logiczne powstałe za pomocą przeprowadzenia odpowiednich operacji logicznych
- ... - dowolna liczba operacji

3 Przykład użycia

3.1 Inicjalizacja zmiennych

- point P1: 1, 3
- num Radius: 2
- circle Circle: P1, Radius
- point P2: 3.1, 6.6
- point P3: 5.6, 10.2
- group Points: P1, P2, P3
- polygon Triangle: Points
- segment Seg: P1, P2
- group Shapes: Circle, Triangle
- iterator I: 5

3.2 Wywołanie transformacji

- fill Square: #red
- draw Shapes[I]

3.3 Zastosowanie instrukcji iteracyjnej

```
loop I start 0 until 4 step 3
    draw Shapes[I]
end
```

3.4 Zastosowanie instrukcji warunkowej

```
check I%2=0
    scale Shapes[I]: 2
    draw Shapes[I]
else check I%3=0
    point Point: 2, 1
    rotate Shapes[I]: 30, Point
    draw Shapes[I]
else
    fill Shapes[I]: #yellow
    draw Shapes[I]
end
```

3.5 Przykładowy program tworzący szachownicę

```
num X: 100
canvas: X, X, #white

num SqrLen: X/8
point P0: 0, 0
point P1: SqrLen, 0
point P2: 0, SqrLen
point P3: SqrLen, SqrLen
group Points: P0, P1, P2, P3
polygon Sqr: Points
fill Sqr: #black

loop I start 0 until 7 step 1
    loop J start 0 until 7 step 1
        check (I+J)%2 = 1
            point P: I*SqrLen, J*SqrLen
            place Sqr: P
            draw Sqr
        end
    end
end
```