Przykładowy algorytm geometryczny (geometria płaska)

Czy odcinki AB oraz CD mają punkt wspólny, czyli czy przecinają się?

A(x1,y1) B(x2,y2) C(x3,y3) D(x4,y4)

następujące cztery **KONIECZNE** warunki muszą być spełnione

max(x1,x2) *większe bądź równe* min(x3,x4)

max(x3,x4) *większe bądź równe* min(x1,x2)

max(y1,y2) *większe bądź równe* min(y3,y4)

max(y3,y4) *większe bądź równe* min(y1,y2)

Przykładowy algorytm geometryczny (geometria płaska)

Dodatkowo z1 i z2 muszą być przeciwnego znaku, lub jedno z nich musi być równe zeru; znaki z1 i z2 określają, czy (x3,y3) i (x4,y4) są na prawo, czy na lewo od wektora AB. Należy wyobrazić sobie wektor AB jako wskazówkę obracającą się wokół punktu A; obrót zgodny ze wskazówkami zegara jest dodatni.

$z1=(x3-x1)(y2-y1) - (y3-y1)(x2-x1)$

$z2=(x4-x1)(y2-y1) - (y4-y1)(x2-x1)$

Inny algorytm geometryczny: ”march of Jarvis”

Algorytm ten (po polsku pochód czy obrys Jarvisa) służy do znalezienia obrysu wypukłego danego zbioru punktów na płaszczyźnie, czyli do znalezienia najmniejszego wielokąta wypukłego zawierającego wszystkie punkty zbioru.

Najistotniejszą częścią algorytmu jest rozpoznawanie, czy dany punkt znajduje się na prawo, czy na lewo od danego wektora (w takim samym znaczeniu jak w poprzednim algorytmie)

(można poszukać w Internecie hasła ”Jarvis march”)

http://edu.pjwstk.edu.pl/wyklady/asd/scb/asd13/main13_p5.html

http://www.cs.unc.edu/~snoeyink/demos/ch/Jarvis.html

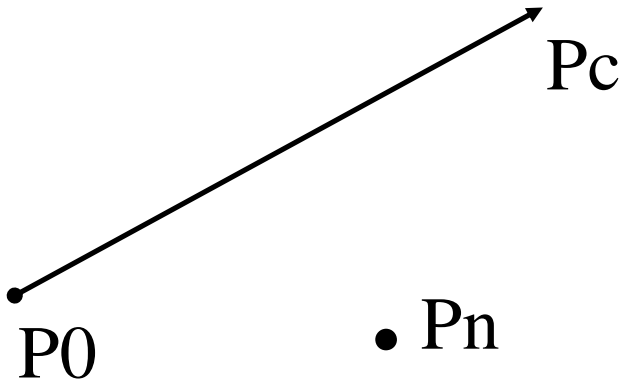
http://personal.kent.edu/~rmuhamma/Compgeometry/MyCG/ConvexHull/jarvisMarch.htm

Inny algorytm geometryczny: ”march of Jarvis”

W algorytmie tym wylicza się czy dany punkt Pn(Xn,Yn) znajduje się na lewo czy na prawo od wektora [P0,Pc] ; początek i koniec wektora mają współrzędne P0(X0,Y0) oraz Pc(Xc,Yc) . Obracamy wektor wokół punktu P0.

Wyliczamy z; jeśli z jest dodatnie to Pn jest na prawo, jeśli ujemne to na lewo.

$$z = (Xn-X0)*(Yc-Y0) – (Yn-Y0)*(Xc-X0)$$



Możliwość „mieszania” języków programowania

W przypadku „rodziny” kompilatorów GNU taka możliwość „mieszania” jest przygotowana. Można np. z wnętrza programu napisanego w języku Fortran wywołać funkcję napisaną w języku C.

Załączony przykład składa się z dwóch plików źródłowych:

przekaz1.f przekaz2.c

(uwaga – kompilator Fortranu np. **f77** albo **f90** albo **gfortran**)

Kompilacja: gcc przekaz2.c –c

Kompilacja: f77 przekaz1.f –c

Linker: f77 przekaz1.o przekaz2.o –o przekaz.exe

Wykonanie: przekaz.exe

Możliwość „mieszania” języków programowania

Kolejny przykład: program w C woła procedurę napisaną w Fortranie.

fnot1.c fnot2.f

(uwaga – kompilator Fortranu np. **f77** albo **f90** albo **gfortran**)

Kompilacja: gcc fnot1.c –c

Kompilacja: f77 fnot2.f –c

Linker: gcc fnot1.o fnot2.o –o fnot.exe

Wykonanie: fnot.exe

Zagadnienia na egzamin

Język C podsumowanie

Struktura programu w języku C

Zmienne, Stałe

Operacje arytmetyczne

Operatory logiczne

Priorytety operatorów

Instrukcje warunkowe (if, ?: , switch)

Pętle (for, while, do-while, instrukcje break i continue)

Macierze i łańcuchy znakowe

(także macierze wielowymiarowe)

Język C podsumowanie

```
 /* Alokuje pamiec na macierz n*n */
double **matrixalloc(int n)
{ int i; double ** bufor;
  bufor = malloc(n*sizeof(double *));
  for(i=0; i<n; i++) bufor[i] = malloc(n*sizeof(double));
  return bufor;
}  /* koniec  */
```

```
....
double **alfa;
alfa = matrixalloc(10);
 ....
```

Język C podsumowanie

Funkcje, prototypy funkcji

Struktury, unie

Rzutowanie na typ

Klasy pamięci zmiennych i funkcji

Zasięg zmiennej, zasięg funkcji

Wskaźniki, macierze, funkcje

Dynamiczna alokacja pamięci (malloc,calloc, sizeof, free)

Argumenty wiersza wywołania programu

Operatory arytmetyczne, logiczne

Operatory bitowe, pola bitowe

Język C podsumowanie

Preprocesor języka C (makra)

Funkcje wejścia/wyjścia(buforowane, niebuforowane, formatowane, bezformatowe)

Typ enum

Instrukcja typedef

Kwalifikatory const, volatile

Czytanie deklaracji obiektów w języku C

Język C podsumowanie

....tego nie było,nie będzie na egzaminie...

biblioteka ncurses

libncurses.a

web.cs.mun.ca/~rod/ncurses/ncurses.html

Koniec zagadnień na egzamin