Obliczenia numeryczne.

W pracach naukowców, konstruktorów i technologów często występuje konieczność obliczania różnorodnych wielkości za pomocą skomplikowanych zależności. Mimo, że wyprowadzone są teoretyczne wzory, z których można coś obliczyć, to okazuje się w praktyce, że nie jest to takie łatwe (np. czy nowowybudowany most zawali się, gdy siła wiatru przekracza 200 km/h). Zamiast bawić się skomplikowanymi matematycznymi zależnościami, układami równań itp., można przecież wykorzystać komputery. **Obliczenia numeryczne** polegały będą na obliczaniu wielkości zdefiniowanych za pomocą zależności matematycznych przy pomocy komputera. Problemy numeryczne można rozwiązywać różnymi metodami. Stosując różne algorytmy można obliczać te zależności szybciej lub wolniej i dawać mogą one wyniki mniej lub bardziej dokładne.

Dokładność obliczeń i kumulowanie błędów.

Nie ma problemu, jeśli posługujemy się liczbami całkowitymi, które w komputerze pamiętane są dokładnie. Z liczbami rzeczywistymi (zwłaszcza okresowymi lub niewymiernymi) komputery radzą sobie już z trudem (zwłaszcza, jeśli mamy do czynienia z dużymi dokładnościami). Liczby rzeczywiste pamiętane są tylko na 6 bajtach (REAL) i dlatego niedokładnie (jak np. zapamiętać liczbę pi?). Liczby rzeczywiste po prostu zaokrągla się i pamiętane są **z określoną dokładnością**, a to powoduje błędy przy dokładnych obliczeniach. Innym komputerowym problemem liczb rzeczywistych jest **kumulowanie błędów.** Przy wielokrotnym powtarzaniu tych samych operacji na liczbach zmiennoprzecinkowych (rzeczywistych), zwłaszcza przy dodawaniu i odejmowaniu, sumowanie błędów powoduje zniekształcanie wyników obliczeń.

Przykładem niech będzie poniższy program:

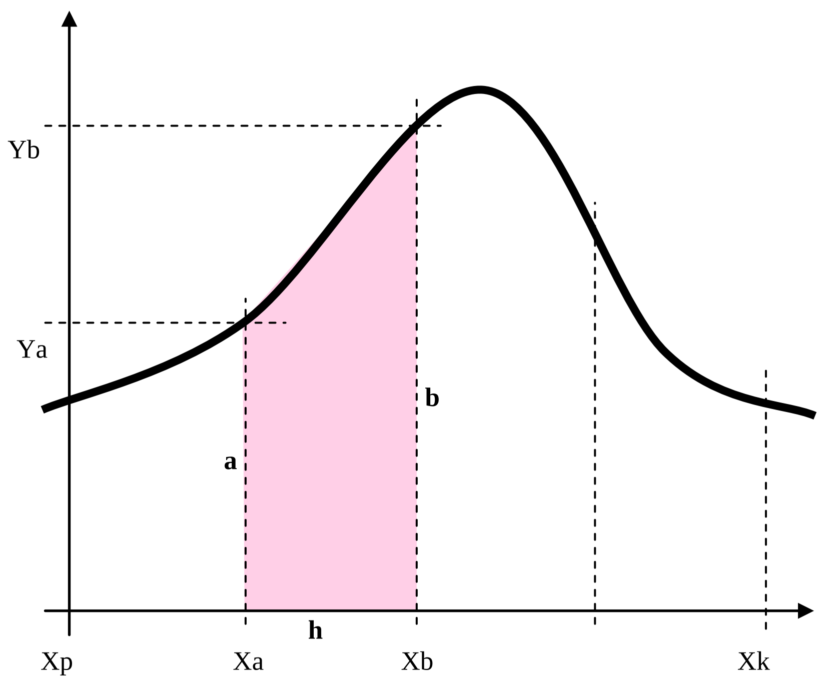
```
suma:=0;
r:=1.1;
for i:= 1 to ile do suma:=suma+r;
writeln(suma/ile);
```

Zsumowanie dużą ilość razy tej samej liczby rzeczywistej (1,1) i podzielenie jej przez ilość sumować powinno teoretycznie dać tą samą liczbę. Kumulowanie się błędów powstałych z zaokrąglania liczb rzeczywistych powoduje, że wyniki przy dużych ilościach są różne. Dla r=1,1 i ile=10 wynik wynosi 1,1 dla ile=1000 wynik wynosi 1,0999999 !

Całkowanie numeryczne (obliczanie pól powierzchni, objętości, długości linii ograniczonych krzywą).

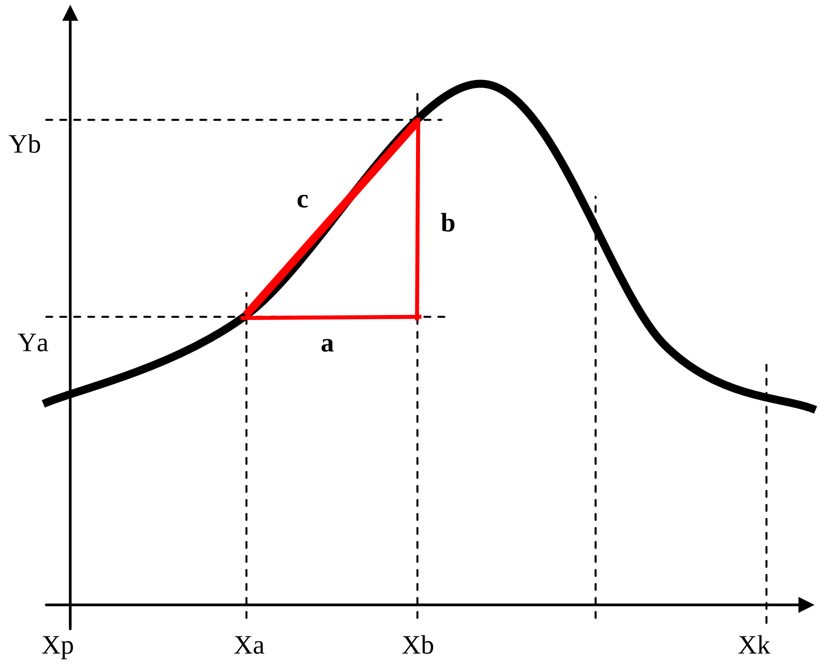
Pole powierzchni

Obliczanie pola powierzchni polega na podzieleniu obszaru pod krzywą na niewielkie fragmenty (np. trapezy), dla których łatwo wyliczyć pole powierzchni. Im więcej trapezów tym dokładniejszy wynik. Jest to tzw. całkowanie numeryczne. Dla niektórych funkcji łatwo jest wyliczyć objętość, czy pole powierzchni za pomocą teoretycznych równań matematycznych (kwadrat, kula itp.), ale skomplikowane funkcje nie są możliwe do rozwiązania metodami teoretycznymi. Dlatego też całkowanie numeryczne ma ogromne znaczenie w wielu dziedzinach nauki i techniki.



Załóżmy teoretycznie, że chcemy obliczyć pole powierzchni pod wykresem w przedziale (Xp..Xk). Dzielimy ten przedział na jednakowej długości odcinki (im ich więcej, tym obliczenia będą dokładniejsze) w naszym przypadku na 4. Pole trapezu obliczamy z wzoru P=h(a+b)/2, na wykresie: P=(Xb-Xa)(Yb+Ya)/2 gdzie:
h - wysokość trapezu - skok na osi X (inaczej mówiąc, jaka jest odległość pomiędzy Xa, a Xb)
a - wartość Y obliczona dla Xa
b - wartość Y obliczona dla Xb
Jeśli zsumujemy pola wszystkich trapezów - otrzymamy (przybliżone) pole pod wykresem.
Jeśli wykres znajduje się pod osią OX - pola mają ujemną wartość, dlatego należy wyliczać je bez znaku (wartość bezwzględna, moduł itp).

Długość krzywej
Idea obliczania długości jest taka sama. Podział osi X, wyliczenie wartości Y. Z twierdzenia Pitagorasa potrafimy wyliczyć wartości przeciwprostokątnych - i je zsumować. Długość pojedynczego odcinka obliczymy z wzoru: $c = \sqrt{a^2 + b^2}$, co przekładając na oznaczenia na wykresie da:
$$D = \sqrt{(Xb - Xa)^2 + (Yb - Ya)^2}$$



Komputerowa realizacja całkowania może być przeprowadzona w różnorodny sposób, ale algorytm postępowania jest zawsze taki sam:

- wartość Xa jest równa Xp
- powtarzaj poniższe czynności
    - wylicz Ya (wstawiając do wzoru na funkcję wartość Xa)
    - wylicz Xb (dodając do Xa SKOK na oso X)
    - wylicz Y (wstawiając do wzoru na funkcję wartość Xb)
    - oblicz pole trapezu (albo długość przeciwprostokątnej)
    - sumuj pola (długości)
    - wstaw do Xa wartość Xb
- aż Xb osiągnie wartość Xk

ĆWICZENIA
- napisz funkcję P_CALKA oraz funkcję D_CALKA, obie z parametrami Xp, Xk, skok, które wyliczą pole powierzchni lub długość krzywej. Podobnie jak w poniższych przykładach, funkcje korzystają z zewnętrznej funkcji Y(x)
- napisz funkcję PR_CALKA, która liczy pole powierzchni, ale za pomocą prostokątów.
- porównaj wyniki działania obu funkcji, jakie są różnice w obliczeniach

W poniższych przykładach całkujemy funkcję $y=-3x^2-4x+5$ w przedziale (-5..5) ze skokiem 0.1
Sposób dochodzenia do wyniku jest identyczny, zmienia się jedynie wzór: pole, długość

POLE

```
var
  Xp,Xk,Xa,Xb,
  Ya,Yb,
  skok,
  pole,
  p:real;

function Y(x:real):real;
begin
  y:=-3*x*x-4*x+5;
end;

begin
  pole:=0;
  Xp:=-5;
  Xk:=5;
  skok:=0.1;

  Xa:=Xp;
  repeat
    Ya:=Y(Xa);
    Xb:=Xa+skok;
    Yb:=Y(Xb);
    p:=skok*(Ya+Yb)/2;
    pole:=pole+abs(p);
    Xa:=Xb;
  until Xa>=Xk;

  writeln(pole);
  readln;
end.
```

DŁUGOŚĆ

```
var
  Xp,Xk,Xa,Xb,
  Ya,Yb,
  skok,
  dlugosc,
  d:real;

function Y(x:real):real;
begin
  y:=-3*x*x-4*x+5;
end;

begin
  dlugosc:=0;
  Xp:=-5;
  Xk:=5;
  skok:=0.1;

  Xa:=Xp;
  repeat
    Ya:=Y(Xa);
    Xb:=Xa+skok;
    Yb:=Y(Xb);
    d:=sqrt(sqr(skok)+sqr(Yb-Ya));
    dlugosc:=dlugosc+d;
    Xa:=Xb;
  until Xa>=Xk;

  writeln(dlugosc);
  readln;
end.
```