```
function [x,fx]=bisekcja(a,b,dokl)
   fa = fun(a)
   while abs(b - a) > dokl
      x = 0.5 * (a + b)
      fx = fun(x)
      test = fa * fx
      if test < 0 then
         b = x
      elseif test > 0 then
         a = x
         fa = fx
      else
         return
      end
   end
endfunction

function f = fun(R)
   f = exp(-0.005*R) ..
       *cos(sqrt(2000-0.01*R*R) ..
       *0.05)-0.01
endfunction
```

Zawartość poprzedniego slajdu zapisuje się np. w pliku `bisect.sci`. Wywołanie w Scilabie ma postać:

```
--> getf("bisect.sci")
--> [x, fx] = bisekcja(0, 400, 1e-5)
 fx  =
    5.549D-09
 x  =
    328.15143
```

Uwaga! Uwzględnienie modyfikacji zawartości pliku `bisect.sci` wymaga ponownego wczytania tego pliku za pomocą `getf`!

Implementacja w języku C

W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery (1994): *Numerical Recipes in C. The Art of Scientific Computing, 2-nd Ed.*, Cambridge University Press

Książka w formacie PDF jest dostępna na stronie `www.nr.com`

```
#include <stdio.h>
#include <stdlib.h>
#include "roots.h"
int main(void) {
  float a, b, root;
  printf("Podaj granice "
         "przedzialu:\n");
  scanf("%f %f", &a, &b);
  if (zbrac(fun, &a, &b) == 1) {
    printf("Granice to %g i %g\n",
                              a, b);
    root = rtbis(fun, a, b, 1.0e-6);
    printf("Pierwiastek to %f.\n",
                              root);
  }
  else {
    printf("Nie mozna znalezc "
           "przedzialu\n");
    printf("Ostatnie granice to "
           "%g i %g\n", a, b);
  }
  system("PAUSE");
  return 0; }
```

Plik `fun.c`

```
#include <math.h>
float fun(float R)
{
   return  exp(-0.005*R)
     *cos(sqrt(2000-0.01*R*R)*0.05)
     -0.01;
}
```

Plik `roots.h`

```
#define FACTOR 1.6
#define NTRY 50
#define JMAX 40
float fun(float);
int zbrac(float (*func)(float),
          float *, float *);
float rtbis(float (*func)(float),
          float, float, float);
```

```
#include "nrutil.h"
#include "roots.h"
int zbrac(float (*func)(float),
          float *x1, float *x2) {
  int j;
  float f1, f2;
  if (*x1 == *x2) nrerror("Bad "
       "initial range in zbrac");
  f1 = (*func)(*x1);
  f2 = (*func)(*x2);
  for (j = 1; j <= NTRY; j++)
  {
    if (f1 * f2 < 0.0) return 1;
    if (fabs(f1) < fabs(f2))
      f1 = (*func)(*x1 += FACTOR
                  * (*x1 - *x2));
    else
      f2 = (*func)(*x2 += FACTOR
                  * (*x2 - *x1));
  }
  return 0;
}
```

```
#include <math.h>
#include "roots.h"
float rtbis(float (*func)(float),
   float x1, float x2, float xacc){
  int j;
  float dx, f, fmid, xmid, rtb;
  f = (*func)(x1);
  fmid = (*func)(x2);
  if (f * fmid >= 0.0)
    nrerror("Root must be bracketed"
          " for bisection in rtbis");
  rtb = f < 0.0 ? (dx=x2-x1, x1) :
                  (dx=x1-x2, x2);
  for (j = 1; j <= JMAX; j++){
    fmid = (*func)(
          xmid = rtb + (dx *= 0.5));
    if (fmid <= 0.0) rtb = xmid;
    if (fabs(dx) < xacc ||
        fmid == 0.0) return rtb;
  }
  nrerror("Too many bisections in"
          " rtbis");
  return 0.0; }
```