Příklad:

```
program dvojkova;
var a:integer;
begin
        readln(a);
        while a > 0 do
        begin
                if a mod 2 = 1 then
                        write(1)
                else    write(0);
                a:=a div 2;
        end;
end.
```

Příklad vylepšený

Při programování je hlavní **myslet, jinak si** často zbytečně **přidělám práci!**

```
program dvojk;
var a:integer;
begin
        readln(a);
        while a > 0 do
        begin
                write(a mod 2);
                a:=a div 2;
        end;
end.
```

Příklad, hledání prvočíselného rozkladu:

```
program rozklad;
var a,i:integer;
begin
        i:=2;
        readln(a);
        while i <= a do
        begin   if (a div i)*i = a then
                begin
                        write(i);
                        a:=a div i;
                end
                else    i:=i+1;
        end;
end.
```

Příklad, hledání prvočíselného rozkladu vylepšený:

```
program rozklad;
var a,i:integer;    opakujeme:boolean;
begin   i:=2;   opakujeme:=false;
        readln(a);
        while i <= a do
        begin   if (a div i)*i = a then
                begin   if opakujeme then
                                write('*')
                        else   opakujeme:=true;
                        write(i);
                        a:=a div i;
                end  else  i:=i+1;
        end;
end.
```

Komentáře

Komentáře umísťujeme do složených závorek:
Příklad: {`Toto je kus komentare`}
Komentář může obsahovat kusy kódu, které si nejsme jisti, zda použijeme:
{`x=sqrt(x); Tak nevim, melo to byt pro odmocninu nebo ne... :-(`}

Pole

- Chceme uskladnit mnoho dat stejného typu (třeba tisíc celočíselných hodnot),
- definuje se v sekci proměnných (tedy `var`)),
- uvádí se klíčovým slovem `array`, následuje definice mezí a určení typu.
- Příklad:
```
var a: array [1..100] of integer;
    soubor:array[5..50] of string;
```
- Do pole přistupujeme operátorem hranatých závorek:
  Příklad:
```
a[1]=10;
soubor[6]='xxx';
{Pozor na:} soubor[1]='meze!';
```

K Eratosthenovu sítu

Jak nagenerovat všechna prvočísla menší než n?

- Naivní algoritmus (generuj a testuj),
- Méně naivní algoritmus generuj a zkoušej dělit jen již nagenerovanými prvočísly.
- Eratosthenes: Nageneruj čísla $2 \ldots n$. Pro i od 2 do $\sqrt{n}$ pokud je i prvočíslo, proškrtej všechny jeho násobky.

Erathostenovo síto

```
program prvoc_do_tis;
var prvocisla: array[2..1000] of boolean;
        i,j:integer;
begin
for i:=2 to 1000 do prvocisla[i]:=true;
for i:=2 to 1000 do
begin
        if prvocisla[i] then
        begin writeln(i,' je prvocislo');
                j:=2;
                while(i*j<=1000) do
                begin
                        prvocisla[i*j]:=false;
                        j:=j+1;
                end;
        end;
end.
```

Algoritmy a složitost

Algoritmus má složitost:

- Statickou – jak velký je jeho zdrojový kód v určitém jazyce (zpravidla nezajímavá),
- dynamickou – jak dlouho běží.
- implicitně máme na mysli složitost **dynamickou!**

Definition

Označuje-li hodnota n délku vstupu, pak složitostí algoritmu $\mathcal{A}$ nazveme funkci f takovou, že pro každý vstup délky n algoritmus $\mathcal{A}$ zastaví po nejvýše $f(n)$ elementárních krocích.

Příklady

Definition

Označuje-li hodnota n délku vstupu, pak složitostí algoritmu $\mathcal{A}$ nazveme funkci f takovou, že pro každý vstup délky n algoritmus $\mathcal{A}$ zastaví po nejvýše $f(n)$ elementárních krocích.

- Erathostenovo síto: lineární vůči testovanému číslu
- Rozklad na prvočísla: Zdánlivě kvadratický vůči dekadickému zápisu, ve skutečnosti lineární.
- Převod do dvojkové soustavy: lineární v délce binárního zápisu.
- Pozor! Složitost počítáme vůči délce zápisu čísla!
- Mínótaurus v bludišti – lineární v počtu hran grafu.
- Stabilní párování (volenkový algoritmus) – nejvýše kvadratické v počtu pánů (resp. dam).

Vyhledávání v poli

- Nesetříděné pole $\Rightarrow$ snadný horní i dolní odhad (prohledat celé pole),
- setříděné pole:
  - unární hledání (listuj jako knihou, dokud nenajdeš),
  - binární hledání (začni uprostřed, pul intervaly),
  - kvadratické vyhledávání, zobecněné kvadratické vyhledávání...

Unární vyhledávání

- Snadný algoritmus, snadná analýza, složitost:
- $\Theta(n)$.

Binární vyhledávání

- Jaká je složitost algoritmu? Kdy je potřeba udělat o krok navíc?
- $\Theta \log n$.