const, const parametry (no)

klíčové slovo const slouží k vytvoření konstantní proměnné, nebo k ochraně proměnných před nechtěnou změnou (především při předávání parametrů ukazatelem nebo referencí)

vytvoření konstanty (neměnné proměnné)
- nelze ji měnit – proměnná označená const je hlídána překladačem před změnou

- konstantní proměnná - obdoba **#define PI 3.1415** z jazyka C
- typ proměnné je součástí definice **const float PI=3.1415;**
- pokud lze, použít const typ místo #define
- obvykle dosazení přímé hodnoty při překladu

- const int a int jsou dva různé/rozlišitelné typy
- při snaze předat (odkazem) const proměnnou na místě nekonstantního parametru funkce dojde k chybě (pozor na překladače vytvářející dočasnou proměnnou)
- volání const parametrem na místě neconst parametru (fce) – nelze
- už je i v C99

const, const parametry (no)

Potlačení možnosti změn u parametrů předávaných funkcím (především) ukazatelem a referencí (odkazem)

```
int fce(const int *i)
int fce1(int const &ii)

double abs(double val);
double abs(const double val); // jiná funkce než předchozí

double fce2(double *pval) {…*pval = 10;…} // změna vně
const int & fce3(double aa)…


double a;
double const ca;

abs(a); // volá se abs(double val)
abs(ca); // volá se abs(double const val)
fce2(&ca); // nelze přeložit;
   // došlo by ke zpřístupnění konstantní proměnné ca ve funkci
```

const, const parametry (no)

shrnutí definicí (typ, ukazatel, reference, const):
T reprezentuje konkrétní typ (int, double, TKomplex, CString …)

`T NázevProměnné`	je proměnná daného typu
`T * NP`	je ukazatel na daný typ
`T & NP`	reference na T
`const T NP` `T const NP`	deklaruje konstantní T (const char a=´b´;)
`T const * NP` `const T* NP`	deklaruje ukazatel na konstantní T
`T const & NP` `const T& NP`	deklaruje referenci na konstantní T
`T * const NP`	deklaruje konstantntí ukazatel na T
`T const * const NP` `const T* const NP`	deklaruje konstatntní ukazatel na konstantní T

const, const parametry (no)

U použití ve více modulech (deklarace-definice v h souboru) – rozdíl v C a C++

- u C je **const char a=´b´;** ekvivalentní **extern const char a=´b´;**
- pro lokální viditelnost – **static const char a=´b´;**
- u C++ je **const char a=´b´;** ekvivalentní **static const char a=´b´;**
- pro globální viditelnost – **extern const char a=´b´;** v .cpp a **extern const char a;** v .h
- pro dodržení kompatibility je tedy vhodné psát včetně modifikátorů extern/static