**Alternativa pro C++**

**/\* testovat zda nacteny symbol je v mnozine symbolu 's1'.**

 **Pokud neni generuje chybu a nacita opakovane ze vstupu**

 **dokud neni nacten symbol z mnozin 's1' a 's2'**

**\*/**

**void test(SYMSET s1,SYMSET s2,int n) {**

**SYMSET pom;**

 **if (!s1[sym]) { /\*sym není v množině s1 \*/**

 **error(n);**

 **nuluj(pom);**

 **sjednot(pom,s1); /\*sjednoti s1, s2 do pom \*/**

 **sjednot(pom,s2);**

 **while (pom[sym] == 0) getsym(); /\* pokud sym není v s1 U s2 cti další \*/**

 **}**

**} // test()**

**void expression(SYMSET fsys) {**

**SYMBOL addop;**

**SYMSET pom;**

 **if ((sym == plus) || (sym == minus)) { /\*unární plus, minus \*/**

 **addop = sym;**

 **getsym();**

 **nuluj(pom);**

 **sjednot(pom,fsys);**

 **pom[plus] = pom[minus] = 1;**

 **term(pom); /\*volame term(fsys U plus U minus ) \*/**

 **}**

 **else {**

 **nuluj(pom);**

 **sjednot(pom,fsys);**

 **pom[plus] = pom[minus] = 1; /\*bez unárního plus minus \*/**

 **term(pom);**

 **}**

 **while ((sym == plus) || (sym == minus)) {**

 **addop = sym;**

 **getsym();**

 **nuluj(pom); /\* iterace {+T} \*/**

 **sjednot(pom,fsys);**

 **pom[plus] = pom[minus] = 1;**

 **term(pom);**

 **}**

**} // expression()**

**void term(SYMSET fsys) {**

**SYMBOL mulop;**

**SYMSET pom;**

 **nuluj(pom);**

 **sjednot(pom,fsys);**

 **pom[times] = pom[slash] = pom[modulo] = 1;**

 **factor(pom); /\*volame factor( followE U follow T \*/**

 **while ((sym == times) || (sym == slash) || (sym == modulo)) {**

 **mulop = sym;**

 **getsym();**

 **sjednot(pom,fsys);**

 **pom[times] = pom[slash] = pom[modulo] = 1;**

 **factor(pom);**

 **}**

**} // term()**

**void factor(SYMSET fsys) {**

**int i;**

**SYMSET pom;**

 **test(facbegsys,fsys,24); /\* test na zacatku faktoru \*/**

 **while (facbegsys[sym]) {**

 **if (sym == ident) {**

 **i = position(id,tx);**

 **if (i == 0) error(11); /\* ten identifikátor není deklarovany \*/**

 **getsym();**

 **} else**

 **if (sym == number) {**

 **if (num > AMAX) {**

 **error(31);**

 **num = 0;**

 **}**

 **getsym();**

 **} else**

 **if (sym == lparen) {**

 **getsym();**

 **nuluj(pom);**

 **sjednot(pom,fsys);**

 **pom[rparen] = 1;**

 **expression(pom);**

 **if (sym == rparen) getsym();/\* follow „lparen expression“ je rparen \*/**

 **else error(22); /\* paren chybi \*/**

 **}**

 **nuluj(pom);**

 **pom[lparen] = 1; /\* pokud ses nezotavil drive, preskoc jen k lparen \*/**

 **test(fsys,pom,23); /\* test na konci faktoru \*/**

 **}**

**} // factor()**