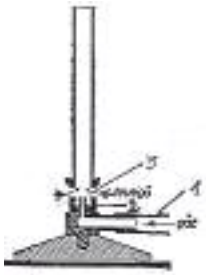
Az 1-es csövön beáramló metán-gáz az elszűkülő 2-es nyíláson (dűzni) nagyobb sebességgel kiáramlik, emiatt a környezetében megnő a dinamikai nyomás és lecsökkenti a sztatikai nyomást, ami szívó hatást fejt ki, és így a külső környezetből a nagyobb légköri nyomás levegőt áramoltat be a gázáramba, ezáltal létrejön egy metán-gáz-levegő keverék, amely a gáz megfelelő égését biztosítja.

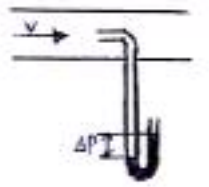
A 3-as nyílás méretét, ahol a levegő beáramlása történik, változtatni lehet, ezáltal szabályozhatóvá válik a gáz-levegő koncentráció és így biztosítható az optimális égési folyamat.



16. ábrán

A Bernoulli-törvény lehetővé teszi, hogy mérőszondák segítségével, folyadék (gáz) áramlási sebességét, térfogat vagy tömeghozamát, és az áramlásban fellépő nyomásokat mérhessük.

A 17. ábra a Pitot-csőnek nevezett mérőszonda elvi vázlatát mutatja be. A nyitott végű manométercsövön leolvasott Δp nyomáskülönbségből kiszámítható az áramlási sebesség :



17. ábra

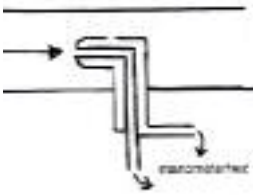
$$v = \sqrt{\frac{2\Delta p}{\rho}} \tag{9}$$

A Pitot-csővel az áramlás térfogat vagy tömeghozama is meghatározható. A térfogathozam : $Q_v = S.v$, és a tömeghozam $Q_m = S.\rho.v$, ahol S az áramlási cső keresztmetszete

18. ábra

A 18. ábrán a Venturi-csőnek nevezett mérőszonda látható. A manométeren mért Δp nyomáskülönbségből az áramlás v sebessége kiszámítható, ennek ismeretében az áramlás hozama is meghatározható:

$$Q_V = S_1 \sqrt{\frac{2\Delta p}{\rho\left(\frac{S_1^2}{S_2^2} - 1\right)}} \tag{10}$$



19. ábra

A Prandtl által kifejlesztett mérőszonda, amely a Pitot- és a Venturi-cső összekapcsolásából alakult ki (Prandtl-cső, 19. ábra), közvetlenül méri a dinamikai nyomást, ennek ismeretében kiszámítható az áramlási sebesség. Szélcsatornákban gázok áramlási sebességének a mérésére leginkább ezt a mérőszondát alkalmazzák.

Puskás Ferenc

Névadási, kódolási konvenciók

A névadási és kódolási konvenciók használata metainformációkat szolgáltat a programok olvasóinak (nem csak írni kell tudni jó programot, hanem olvasni is tudni kell őket – hibajavítás, későbbi módosítások stb. érdekében).

Az utasítások, alaptípusok stb. általában adottak egy programozási nyelvre nézve, így a programozó általában csak a felhasználói típusok, konstansok, változók stb. neveit adhatja meg, vagyis új azonosítókat vezethet be a programokba.

Az első, legfontosabb kérdés az, hogy a fordítóprogram különbséget tesz-e a kisbetűk és a nagybetűk között (case sensitivity), ha különbséget tesz, akkor igazodnunk kell a fordítóprogram íróinak elképzeléseihez a program megírásánál, ellenkező esetben már a fordítás sem történhet meg helyesen (például a Pascal nem tesz különbséget, de a C különbséget tesz kis- és nagybetűk között).

Az azonosítók deklarálásánál figyeljünk arra, hogy az adott nevek minél beszédesebbek legyenek, ne legyenek túl rövidek, de túl hosszúak sem. A forráskód későbbi újraolvasásánál, javításánál nem sokat mondanak az a, b, c, d, e, f nevű változók, de az EzEgyEgészVáltozóAHarmadikForCiklusSzámára név is elég zavaró lehet.

Követendő általános elvek:

- Az adott neveknek legyen jelentése, használjunk beszédes neveket. A változónevek rövidek, de sokatmondóak legyenek. A változónevekből a használatukra lehessen következtetni. Egykarakteres változónevek használatát általában mellőzni kell.
- Típusok, osztályok, változók deklarációjánál ha a név több szóból áll, minden szót kezdjünk nagybetűvel, a szavak között ne hagyjunk sem szóközt, sem aláhúzásjelt („_"), pl.: *IskolaAzonosítóKód*. A beépített alaptípusokat írjuk kisbetűvel: *byte, integer, string*.
- A konstansokat szedjük általában nagybetűkkel, itt a szavak között – ha több szóból áll a neve – használjunk aláhúzásjelt („_"), pl. *MAX_INT*.
- Eljárások, függvények neveire használjunk igéket, melyek leírják a cselekvést. pl. *Nyomtat*, *Rajzol*. A paraméterek, visszatérési értékek nevei is legyenek beszédesek és írják le a paraméter jelentését – használjunk erre a célra főneveket. Az eljárások, függvények neveiben pontosítsuk a feladatkört is pl. *SaveToFile*, *SaveToStream*.
- Rekordok, struktúrák esetében a mezőneveket kezdjük kisbetűvel, ha több szóból állnak, a második szótól kezdődően minden szó nagybetűvel kezdődjön:

```
TSzemely = record
  csaladNev: string[20];
  szemelyNev: string[20];
  eletKor: integer;
end;
```

- A ciklusváltozókat mindig ugyanazzal a névvel lássuk el programjainkban: *i, j, k*. Ha háromnál több beágyazott ciklust használunk, akkor a ciklusváltozók nevei legyenek beszédesek.
- A globális változók neveit lássuk el a *g* előtaggal. Az ideiglenes, temporális változók neveit lássuk el a *tmp* előtaggal.

A forráskódot a jól olvashatóság érdekében lássuk el megjegyzésekkel is. Különösen vonatkozik ez a típusok, változók, konstansok, algoritmusok elő- és utófeltételei, bizonyos megkötések stb. megadásakor. Egyes programozási nyelvekben ismert a dokumentációs megjegyzés fogalma is, amelyeket összegyűjtve, az automatikus dokumentációgenerátor jól használható programozási dokumentációt tud előállítani. A forrásszövegekbe beírt megjegyzések az első lépések egy jó dokumentáció elkészítéséhez.

A forráskód kinézete, szerkesztése, a fehér karakterek használata is figyelemreméltó. Lehetőleg olvashatóan határoljuk el a blokkokat, hogy mindig tudjuk mire is vonatkozik az adott utasítás. A blokkokon belül használjunk bekezdéseket, de egy sor hossza ne legyen túl nagy. Számos programozási nyelv kötött sor-formátummal dolgozik (pl. első

három karakter a címke, utána szóköz, utána utasítás, szóköz, operandusok stb.), de a nyelvek nagytöbbsége kötetlen programírást biztosít.

Kövessünk végig egy pár programozási nyelvet, milyen névadási, kódolási konvenciók használatosak bennük:

Borland Dephi

A típusok, s így az osztályok nevei is „T" betűvel kezdődnek, az inerfészeké pedig „I" betűvel, a kivételeké „E"-vel. A *private* mezők nevei „f"-fel kezdődnek. A felsorolt típusok elemei általában a típus nevéhez igazodnak, előtagként tartalmazzák a típusnév szavainak kezdőbetűit:

```
TLineStyle = (lsNone,lsDoted,lsDashed,lsSolid);
```

Típusok és osztályok:

Elemek	*Előtag*	*Példa*
Kivétel	„E"	`EMyError = class(Exception)`
Osztályok, típusok	„T"	`TMyClass = class(TObject)`
Interfész	„I"	`IUnknown`
Mezők (rejtett)	„f"	`fVisible`
Események	„On"	`OnMouseDown`

Változók:

Típus	*Előtag*	*Példa*
string	„s"	`sName`
boolean	„b"	`bIsGood`
integer	„i"	`iNumber`
pointer	„p"	`pMyPointer`
DateTime	„dt"	`dtBirthday`
Currency	„cur"	`curSallary`

Komponensek

Típus	*Előtag*	*Példa*
Form	„frm"	`frmMain`
Button	„btn"	`btnOK`
Label	„lbl"	`lblName`
Edit	„ed"	`edPassword`
ComboBox	„cb"	`cbFont`
ListBox	„lb"	`lbFiles`
Table	„tbl"	`tblMaster`
Query	„qry"	`qryTeachers`
DataSource	„ds"	`dsSchool`
DataBase	„db"	`dbMyDataBase`
PaintBox	„pb"	`pbMyPicture`
MediaPlayer	„mp"	`mpMP3Player`
OpenDialog	„OpenDialog"	`OpenDialog`
CloseDialog	„CloseDialog"	`CloseDialog`

C, C++, C#

Ezekben a programozási nyelvekben a Simonyi Károly által bevezetett *magyar stílusú* jelölést *(Hungarian Notation)* használjuk. Az egyes változók elnevezésére nem rövid és értelmetlen betűszavakat használunk, nem is hosszú magyarázkodó nevet, hanem olyan azonosítókat, amelyekben a név első része az adattípust, második része az adat jelentését mutatja:

Típus	*Előtag*	*Példa*
logikai	„b”	`bool bIsGood;`
karakter	„c”	`char cLetter;`
C++ sztring	„str”	`string strName;`
rövid egész	„si”	`short siChairs;`
egész	„i”	`int iNumber;`
hosszú egész	„li”	`long liStars;`
lebegőpontos	„f”	`float fPercent;`
dupla pontosságú	„d”	`double dMiles;`
hosszú dupla	„ld”	`long double ldLightYears;`
Null-terminál sztring	„sz”	`char szName[NAME_LEN];`
Input File Stream	„if”	`ifstream ifNameFile;`
Input Stream	„is”	`void fct(istream &risIn);`
Output File Stream	„of”	`ofstream ofNameFile;`
Output Stream	„os”	`void fct(ostream &rosIn);`
struktúra	„S”	`struct SPoint {`
osztály	„C”	`class CPerson {`
struktúra példány	a struktúra neve vagy rövidítése	`SPoint pointLeft;` `SPoint ptLeft;`
objektum	az osztály neve vagy rövidítése	`CPerson personFound;` `CPerson perFound;`

Típus	*Elő-előtag*	*Példa*
előjel nélküli	„u”	`unsigned short usiNumber;`
konstans paraméter	„k”	`void p(const long kliNr)`
referencia paraméter	„r”	`void p(long &rliNr)`
statikus	„s”	`static char scChoice;`
tömb	„rg”	`float rgfTemp[MAX_TEMP];`
tagváltozó, metódus	„m_”	`char m_cLetter;`
függvény	„fn”	`char fncLetter();`

Típus	*Elő-előtag*	*Példa*
mutató	„p”	`char *pcGrade;`
közeli mutató	„np”	`char *npcGrade;`
távoli mutató	„lp”	`char *lpcGrade;`
tömb	„a”	`int aiVect[];`
dinamikus tömb	„prg”	`char *prgcGrades;`

Más előtagok: byte: „by”, word: „w”, szám vagy intervallum: „n”, valós szám: „r”.

Java

A lokális változók inicializálása lehetőleg a deklarálásnál történjen meg. Ettől csak akkor tekinthetünk el, ha a változó kezdőértéke először valamiféle kiértékelést igényel.

Deklarációt csupán blokkok elejére tegyünk. Blokknak tekintünk ebben az esetben kapcsoszárójellel határolt kódrészeket. Ne várjunk a változó deklarálásával az első használatig. A kevésbé tapasztalt programozó összezavarodhat, és hátráltatja a kód hordozhatóságát.

A *for* ciklusok változóit a cikluson belül deklaráljuk:

```
for (int i = 0; i < maxLoops; i++) { ... }
```

Java osztályok és interfészek kódolásánál a következő formázó szabályokat kell szem előtt tartani:

- Ne legyen szóköz a metódus neve és a paraméterlista kezdő „(” között.
- Nyitókapocs „{” ugyanannak a sornak a végén van, amelyben a deklaráció.
- Zárókapocs „}” új sort kezd, a nyitókapocsnak megfelelő szintre van rendezve, kivéve, ha null utasításról van szó. Ilyenkor közvetlenül a nyitókapocs után áll.

Utasítások írásakor a következő szabályokat tartsuk be:

- A bennfoglalt utasítások egy szinttel beljebb legyenek rendezve.
- A nyitókapocsnak az összetett utasítás kezdősorának végén kell lennie, a zárókapocs sor elején van, és az összetett utasítás elejéhez van igazítva.
- Minden egyes, még magában álló utasítás körül is kapcsok vannak, ha azok egy irányító struktúra, mint pl.: *if-else* vagy *for* utasítás részei.

Típus	*Névadási konvenciók*	*Példa*
package	Egy package-név első komponense csak kis ASCII karaktereket tartalmazhat, és vagy a legfelső szintű domain-nevek (com, edu, gov, mil, net, org), egyike, vagy egy az ISO 3166, 1981 szabvány által specifikált kétbetűs angol ország-azonosító (hu, ro, de, at). A package-név további komponensei a cég belső névadási szokásait tükrözik.	`com.sun.eng`
osztály	Neveik főnevek. Összetett esetben minden egyes tag kezdőbetűje nagy.	`class Datum;`

Típus	*Névadási konvenciók*	*Példa*
interfész	Hasonlóan az osztályokhoz.	`interface Adat;`
metódus	A metódusok nevei kisbetűs igék. Összetett esetben a második tagtól a tagok nagybetűvel kezdődnek.	`run();` `runFast();`

változó	A változók nevei kisbetűsek. Összetett esetben a második tagtól a tagok nagybetűvel kezdődnek. Változónevek soha nem kezdődhetnek aláhúzással („_”), sem dollárjellel („$”) még ha mindkettő szintaktikailag engedélyezett is.	`int i;`
		`char c;`
		`float myWidth;`

konstans	Nagybetűs szavak, közöttük aláhúzás.	`MIN_WIDTH = 4`

Kovács Lehel

A magyar kémiai szaknyelv kialakulásáról

A XVIII. század második feléig a tudományos világban a latin nyelv volt a kommunikáció lehetősége. Ez volt az oka, hogy a magyar nyelv nagyon szegényes volt a természettudományok terén. A nagy nemzetek (francia, német, angol) már valamivel hamarább kezdték nemzeti nyelvüket használni, de valójában csak a polgári fejlődés vonta maga után a nemzeti nyelvek megerősödését A nyugati kultúra magyarországi és erdélyi terjedése feltételezte az anyanyelvi kultúra kialakulását. Könyvfordításokkal próbálkoztak, de a kémiai tárgyúaknál nagy nehézséget jelentett, hogy a magyar nem rokon nyelv a nyugatiakkal, s ezért nem léteztek a rokon kifejezések. Így például a fémek közül is csak ötnek volt magyar neve (vas, réz, arany, ezüst, kéneső – a higany régi neve, amely a kömösü török szóból ered), míg a nyugati világ többet ismert. A nemfémek közül csak a kén és a szén neve ősi. A bányászatban használatossá vált kémiával kapcsolatos kifejezések általában német hatásra torzított nevek voltak, mint pl. antimon-piskolc, borax-póris, arzén-rozsnika. Az orvosok, gyógyszerészek próbálkoztak köznép számára érthető szövegek magyarnyelvű kiadásával. Ezek közül legjelentősebb Mátyus Istvánnak (1725 – 1802) 1762-ben Kolozsváron kiadott Dietetica című műve, melyben orvosi, egészségügyi kérdések mellett gyógyvizekkel és ezek elemzésével is foglalkozott. Ebben közölt először magyar nyelven kémia jellegű szöveget, amely a mai olvasónak nem nagyon érthető, furcsa hangzású. Igazolja ezt egy idézet:

„…Ha Gálitzkő olajtól vagy spiritustól erősen felbuzdul, egyéb gyengébb savanyuktól is…a viola Juleptől meg-zöldül…savanyuság ellen való fejér föld és húgy ízű só vagyon” (mai értelmezése: ha kénsav vagy gyengébb savak hatására pezseg és az ibolya-főzet indikátort zöldre változtatja, kalcium-karbonát és szóda van jelen). Ez időben a köznapi gyakorlatban az orvosok, gyógyszerészek írtak magyar nyelven. Példaként álljon itt egy állatok kezelésére leírt beszámoló másolata 1787-ből. *(lásd a mellékelt képen)*