19. Függelék: Számítástechnika és analízis

Simonovits Miklós

19.1. Bevezetés a Függelékhez

A függelék célja, hogy rámutassunk arra, hogyan használható a számítógép

(a) az analízisbeli fogalmak *szemléltetésére*,

(b) analízisbeli feladatok *megoldásának megsejtésére*.

Az (a)-ban tehát elsősorban arról van szó, hogy egy definícióra vagy tételre gyártunk könnyen illusztrációkat, míg (b)-ben arról írunk, hogy ha adott egy feladat, amelynek nem tudjuk a megoldását, akkor a megoldás megsejtéséhez kísérleteket végzünk.

A gép jól használható a számelméletben, analízisben, fizikában: kis munkával hatékony programokat írhatunk. Más területeken, pl. kombinatorikában a programok megírása több gondot jelent, és kevésbé látványos az eredmény.

Itt az **analízis illusztrálására szorítkozunk**. Ezen belül elsősorban a

1. határérték megsejtetésével,
2. függvények ábrázolásával, elemzésével,
3. polinomapproximációval foglalkozunk.

A **számítástechnika oldaláról** az alábbiakat vizsgáljuk meg:

1. Hogyan és mire használhatunk kész programokat?
2. Mikor kell komolyabb programcsomagokat használni?
3. Mire használhatunk magunk által írt egyszerű programokat?
4. Mely fogalmaknál hasznos, ha számítógépet használunk?
5. Hogyan csap be a számítógép, ha nem vigyázunk?

Mindezt elsősorban a matematika oldaláról, másodsorban a számítástechnikában a QBasic programnyelv és a Maple matematikai programcsomag segítségével vizsgáljuk meg[73].

Nem célunk itt a számítógépes ismeretek taglalása. Amit a számítástechnikáról leírunk, az inkább csak emlékeztető. Célunk, hogy megszerettessük a számítógép használatát, és megválaszoljuk a fenti kérdéssort. Filozófiánk, hogy nagyon kevés számítástechnikai ismerettel is nagyon sok dolgot kiszámolhatunk a tudományokban. Ugyanilyen szemléletű, de sokkal részletesebb anyag található Fried Katalin és Simonovits Miklós [6] könyvében. Az itteni programok, ezen fejezet egy bővebb változata, és bizonyos kiegészítő további anyagok letölthetők Simonovits Miklós honlapjáról, a

www.renyi.hu/~miki/Szamtech2.html (237)

címről.

Feltételezzük, hogy az Olvasó programjaink megszemléléséből mindent megért, ami egy egyszerűbb program megírásához szükséges, vagy ha nem, akkor letölti a szükséges

[73] Céljainkra a Basic teljesen elegendő, de egy kis kitekintést beiktattunk a Pascal felé is.

,,dokumentációt" az INTERNET-ről, beütvén kedvenc keresőjébe, hogy ,,BASIC program" wagy ,,PASCAL program", vagy ,,MAPLE program",...

Angol vagy magyar?

Szerintem egy matematikus, programozó, vagy természettudományt tanuló egyetemistának okosabb a programok angol verzióit használnia. Az alábbi leírások is ehhez igazodnak. De megadjuk az általunk használt legfontosabb angol szavak egy rövid listáját.

basic	**alap(vető)**	**line**	**vonal**	**save**	**(el)ment**
black	**fekete**	**maple**	**kanadai tölgy**	**screen**	**képernyő**
color	**szín**	**menu**	**menü**	**solve**	**megold**
display	**képernyő**	**next**	**következő**	**square**	**négyzet**
end	**vége**	**notation**	**jelölés**	**step**	**lépés/lépcső**
error	**hiba**	**option**	**választási lehetőség**	**substitute**	**behelyettesít**
evaluate	**kiértékel**	**plot**	**kirajzol**	**then**	**akkor**
exit	**kiszáll, abbahagy**	**print**	**nyomtat**	**tools**	**eszközök**
free	**szabad, ingyenes**	**quit**	**kiszáll, abbahagy,**	**untitled**	**címnélküli**
help	**segítség (súgó)**	**read**	**olvas**	**worksheet**	**munkalap**
infinity	**végtelen**	**restart**	**újraindít**	**write**	**ír**
input	**bemenet**	**return**	**visszatér**		
limit	**limesz, határ**	**run**	**fut**		

Milyen matematikai programcsomagokat használjunk?

A címben feltett kérdésre sokféle válasz adható, és remélhetően mindegyik válasz gyorsan avul el. Az alábbiakat érdemes figyelembe venni:

1. Ha *egy* ilyen programot megtanulunk, utána bármelyik másikat már sokkal könnyebben tanuljuk meg. Ilyen értelemben beszélhetünk nemelavuló tudásról. Ilyen szempontból is mindegy, hogy MAPLE-t vagy MATHEMATICA-t használunk-e. Eléggé elterjedt programcsomag a MATLAB, de az másra való. [74]
2. Majdnem mindig olyan programot érdemes használni, amelyiket a környezetünkben mások is használnak, és amelyikről megkérdezhetünk valakit, ha elakadunk. Ez *sokkal fontosabb mint gondolnánk.*
   (a) Vannak, akik szakkönyvek alapján szeretik elsajátítani a komputerprogramok használatát. Ilyen emberből is kétfajta van: az egyik mindig csak a szükséges minimumot olvassa el. Ha elakad, leemel egy könyvet a polcról, kikeresi, amire szüksége van, majd visszarakja a könyvet a helyére. A másik ember előbb átolvas egy szakkönyvet, és csak utána kezd dolgozni a programmal.
   (b) Sokan leülnek mások mellé, őket megfigyelve elkezdenek egyedül is próbálkozni. Ha elakadnak, megkérdeznek valakit, majd folytatják a munkájukat.

[74] Lásd pl. http://en.wikipedia.org/wiki/MATLAB

Ahhoz, hogy eldöntsük, melyik programot használjuk, *ismernünk kell magunkat, a környezetünket* és az elérhető könyveket. Eszerint kell választanunk.

3. Két általános célú matematikai programcsomagot ajánljuk az Olvasó figyelmébe:
   (a) MAPLE: A Waterlooi Egyetemen (Kanada) fejlesztették ki.[75] Mindent tud, amire egy diáknak vagy egy matematikusnak szüksége lehet.
   (b) MATHEMATICA: Stephen Wolfram kifejlesztett programcsomag, kb. ugyanazt tudja, mint a MAPLE.[76]

[46]


(46)
Miki: Valakinek ellenoriznie kell majd a punktuaciomat, helyesirasomat

A BASIC ill. PASCAL beszerzése

LINUX-ban minden ingyen, legálisan megszerezhető, így könnyen és ingyen feltehető a PASCAL és a BASIC is. Itt két PASCAL verziót említünk meg, a FREE PASCAL-t és a GNU PASCAL-t. BASIC-ből használhatjuk a BASIC 256-ot. De az INTERNET-en ezekhez leírást is kell keresnünk. [47] Windows-ban a QBASIC ingyen letölthető. A nagy programcsomagok, (pl. a MAPLE) ott vannak az egyetemi laborokban.


(47)
Miki: rakjam ki a home-page-emre?

Amit itt a PASCAL-ról írunk az elsődlegesen FREE PASCAL-ban lett tesztelve, illetve, a Windows-os TURBO PASCAL-ban, a BASIC programok pedig QBASIC-ben.

Melyik programnyelvet használjuk?

Majdnem mindegy: használjuk azt, amelyik a legszimpatikusabb. A mi feladatainkra a BASIC is teljesen megfelel. Mi ennek egy „nyelvjárását", a QBASIC-et használjuk: könnyű megtanulni és könnyű benne rajzolni.

Választhattuk volna a PASCAL programnyelvet is, de – ami a PASCAL-t illeti – a számunkra szükséges nagyon egyszerű programoknál a QBASIC használata a legegyszerűbb. Emellett a programok átírása PASCAL-ra majdnem automatikus. A Simonovits Miklós honlapon találunk néhány, az itt szereplő BASIC programokkal ekvivalens PASCAL programot. Ide is betettem két PASCAL programot.

Mikor használjunk MAPLE-t, mikor BASIC-et, mikor PASCAL-t?

Itt nem az a kérdés, hogy a QBASIC jobb-e, vagy a MAPLE, hanem, hogy mikor használjunk olyan programot, amelyik azonnal megadja a választ, és mikor írjunk valamilyen programnyelvben saját programot. Az, hogy ezt a programot BASIC-ben, PASCAL-ban, MAPLE-ben, vagy C-ben írjuk meg, másodlagos. Használjuk azt, amelyik szimpatikus számunkra, és amelyik nem vonja el a figyelmünket a jelenségről. A programfutás gyorsasága itt lényegtelen, a programírás egyszerűsége viszont fontos.

Ha egy függvényt, vagy felületet ki szeretnénk rajzolni, a MAPLE ebben kitűnő. Többnyire a BASIC-ben lényegesen több időre lesz szükségünk egy hasonlóan hasznos

[75] Elnevezése is erre utal: MAPLE a kanadai tölgy-juhar, melynek levele Kanada szimbóluma.

[76] Ha egy matematikai fogalmat szeretnénk gyorsan megtalálni az INTERNET-en, az első helyeken a Wikipedia és a Wolfram World szokott bejönni...

ábrához. Gyors, felületes vizsgálatra a MAPLE egyértelműen jobb, mint a saját programunk. Pontos és gyors választ kapunk a jól feltett kérdéseinkre. [77] Ilyenkor válasszuk a MAPLE-t.

Az analízisben a saját programjaink rövidek, és gyakran több információt adnak a jelenségről, mint a kész programok.

Mikor használjunk PASCAL-t BASIC helyett? Erre bonyolultabb a válasz. Ha nem akarunk rajzolni, de a megírandó program kissé összetettebb, akkor a PASCAL-lal jobban járunk. Ha csak egy egyszerűbb rajzot akarunk elkészíteni, akkor a válasszuk a QBASIC-et. Windows-ban, a Turbo Pascal-ban sem nehéz rajzolni. Néhány példát mutatunk erre a (237)-en. De ennél mélyebbre itt nem megyünk a kérdés diszkussziójában.

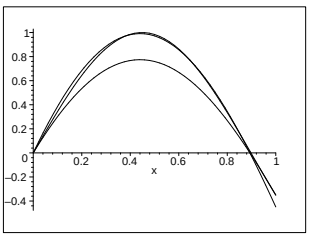
A kötelező óvatosság

Három dologra vigyázzunk:

(i) A gép becsaphat. (ii) Adhat olyan választ, amivel semmit sem tudunk kezdeni. (iii) Értelmetlenül elpocsékolhatjuk az időnket, ahelyett, hogy egy kis gondolkodással megoldanánk a problémánkat.[78]

Részletesebben:

(i) Például, ha egy függvényt szeretnénk számítógéppel megsejteni (és konkrét értékeit ki tudjuk számolni), ez nehéz akkor, amikor a keresett függvény nagyon közel van egy másik, sokkal egyszerűbb függvényhez. Ezért számítógéppel juthatunk helyes és helytelen eredményre egyaránt. Kell ahhoz némi gyakorlat, hogy tudjuk, mikor bízhatunk meg egy géppel kapott eredményben. A 19.1. ábrán a $\sin(\pi x)$ és egy 6-odfokú polinomközelítése látható, pontosabban 3 görbe: legalul $\sin x$ egy 6-odfokú approximációja[79], $B_6(x)$, felette $\sin x$ és $1.29B_6(x)$, de $\sin x$ es $1.29B_6(x)$ grafikonjai közel vannak egymáshoz. Még az sem világos, hogy melyik grafikon melyik függvényhez tartozik. Mégse higgyük, hogy a $\sin(\pi x)$ az adott intervallumon egy polinom.



19.1. ábra: $\sin(\pi x)$ és egy közelítése

(ii) Amikor használni kezdjük a gépet, tudnunk kell, milyen formában keressük a választ. Ha pl. az ikerprímek iránt érdeklődünk, nem kérdezhetjük meg, hogy van-e végtelen sok ikerprím, de könnyen írhatunk olyan programot, amelyik az 1000000 alattiakat kiírja.

(iii) Ha nem vigyázunk, sok időt töltünk el egy program megírásával, és a gép olyan választ ad, amivel semmit sem tudunk kezdeni. Pl. kiírja ezeket a prímeket. Jobban járunk, ha azt íratjuk ki, milyen sűrűn talál ikerprímeket.

[77] Akkor is kitűnő segítség a MAPLE ha előadást tartunk és görbéket, vagy felületeket szeretnénk kirajzoltatni.

[78] Ez nem vonatkozik erre a tanulási folyamatra!

[79] Ez itt éppen a korábban, a 214. oldal, ill. a (243) képlet approximációja.

Egy adott feladat megoldását segítő, saját magunk által írt program egyaránt lehet hasznos és káros. Felhívhatja a figyelmünket arra, amit egyébként nem vagy csak nehezebben vennénk észre. Másrészt elterelheti a figyelmünket valami fontosabbról, leblokkolhatja a gondolatainkat, nagyon sok időnket elviheti. Próbáljunk megmaradni az arany középúton.

A fejezet felépítése. Az első részben a QBASIC programok alkalmazásáról írunk, majd teszünk egy rövid kitérőt a PASCAL programokra, a másodikban a MAPLE szempontjából vizsgáljuk a számítástechnika felhasználását. Ennek megfelelően az alfejezetek lényegében a következőek:

ÁLTALÁNOS KÉRDÉSEK	BASIC	MAPLE
Melyik nyelvet?	KEDVCSINÁLÓ	KEDVCSINÁLÓ
Melyik programcsomagot?	Határértek megsejtése	Határérték megsejtése
Hogyan csap be a számítógép?	Konvergenciasebesség	Konvergenciasebesség
Programátírás PASCAL-ra	Newton algoritmus $\sqrt{a}$-ra	
A programstruktúra fontossága	π Archimedeszi közelítése	
	$n!$ Stirling közelítése	
	Sorok összege	
	Integrál közelítése	
	Függényábrázolás	Függvényábrázolás
	Függények vlzsgálata	Függvények vizsgálata
	HOGYAN FOLYTASSUK?	HOGYAN FOLYTASSUK?
		Polinommal közelítés

Bevalljuk, azért vettük előre a QBASIC-et, mert a MAPLE olyan imponálóan sokat tud, hogy utána az olvasó esetleg indokolatlanul nehézkesnek képzelné a QBASIC-et.

19.2. BASIC programok: Kezdő lépések

Rövid összefoglaló a BASIC-ről

Ha elindítjuk a QBASIC programot, akkor megjelenik a QBASIC integrált környezet, vagyis egy olyan editor, amelyikben programokat írhatunk és azokat kiírhatjuk lemezre, továbbfejleszthetjük, futtathatjuk.[80] Közepes bonyolultságú programoknál a hibákat is könnyen megtalálhatjuk a QBASIC-ben.

Ez nem egy számítástechnika-tankönyv. Csak nagyon kevés BASIC kulcsszót használunk, és a számítástechnikai csiszolásokat majdnem teljesen elhagyjuk. Csak annyira írjuk le a BASIC kulcsszavak jelentését, amennyire szükségünk lesz rájuk.

[80] Ezzel szemben van olyan programnyelvhasználat is, ahol egy editorban írjuk a programot és egy másik helyen ,,kompiláljuk", majd a ,,kompilált programot" futtatjuk.

CLS SCREEN 12 COLOR(K) LINE(x,y)–(u,v)	Clear Screen: letörli a képernyőt. Grafikus képernyő, rajzoláshoz, finom felbontás: 640×480 tükrözött koordináta-rendszer. Beállítja a színeket, pl. K = 4 piros, 1 kék. Egyenesszakasszal köti össze a két pontot.
INPUT x PRINT x	Bekéri az x változó értékét. Kinyomtatja az x változó értékét.
FOR i = 1 TO n ... NEXT i	Végrehajtja az összetartozó FOR és NEXT közötti parancsokat a ciklusváltozó kijelölt értékeire.
IF <felt> THEN <para>	Az IF utáni feltételt ellenőrzi, és teljesülése esetén a THEN utáni parancsokat végrehajtja.
REM	Feljegyzésekre szolgál, ami a sorban utána van, a program futását nem befolyásolja.
END	Leállítja a programot.
a^b SQR(x)	Hatványozás: a^b Négyzetgyökvonás: $\sqrt{x}$

19.3. Kedvcsináló QBasic programokhoz

19.3.1. Sorozatok szemléltetése, határértéke

Tegyük fel, hogy $\lim \sqrt[n]{n}$-re vagyunk kíváncsiak: létezik-e, és ha igen, mennyi? Írjunk ennek vizsgálatára programot.

Tekintsük az alábbi programot:

1. Program **(Sorozat határértéke).**

```
PRINT "n = "
INPUT n                      n bekérése  ||  (i)
a = n^(1/n)                    számolás  ||  (s)
PRINT a                        eredmény  ||  (e)
```

Forma: Az itt szereplő programokat többnyire a fenti formában adjuk meg: a szedésénél három oszlopot használunk. A baloldaliba teszük magukat a programsorokat. Csak ez számít. A ||-tól jobbra hivatkozásokat fogunk írni, középre némi magyarázatot, amit azután kicsit alaposabban is elmagyarázunk.

Keretprogramok: programjainknak gyakran van egy soruk, – itt a (s) – ahova tetszőleges képletet írhatunk, és akkor a program az ennek megfelelő eredményt adja ki.

Magyarázat: A program a 2. sorában bekéri n értékét. Az 1. sor emlékeztet, hogy n értékét kéri, a 3-adikban n-edik gyököt von belőle, a 4-edikben kiírja az eredményt. Ha beadjuk $n = 1$, 2, 3, 4, 5, 6, 100, 1000, 10000-et, rendre a következőket kapjuk:

$\sqrt[n]{n}$:	1	2	3	4	5	6	100	1000	10000
	1	1.414	1.442	1.414	1.379	1.348	1.047	1.00693	1.00092

Ebből megsejthetjük, hogy a sorozat 1-hez tart.

19.3.2. Egy rekurzió határértéke

A rekurziók a kezdők számára kissé nehezen tekinthetők át. Ilyenkor egy egyszerű programocska valóban sokat segíthet. Nézzük pl. a 60. oldalon található 4.1. Példa (15) sorozatát:

$$a_1 = 0, \qquad a_{n+1} = \sqrt{2 + a_n}.$$

Konvergens-e? Mihez tart?

2. PROGRAM **(Rekurziós határérték).**

```
INPUT "Meddig irjam ki?  n = "; n
a = 0                                        inicializálás ‖ (i)
FOR i = 1 TO n                                ciklus eleje ‖ (e)
a = SQR(2 + a)                                  a rekurzió ‖ (r)
PRINT i, a                                        kiíratás ‖
NEXT i                                         ciklus vége ‖ (v)
```

Itt egy újabb számítástechnikai fogalmat, a **for-next** ciklust vezettük be. Az (e)-(v) sor-pár azt intézi el, hogy a közöttük levő sorokat a program végrehajtsa $i = 1, \ldots, n$-re. Ha a programot $n = 14$-gyel futtatjuk, az alábbit kapjuk:

1	2	3	4	5	6	7	11	12	14
1.41	1.85	1.96	1.990	1.997	1.9994	1.99985	1.999999	2	2

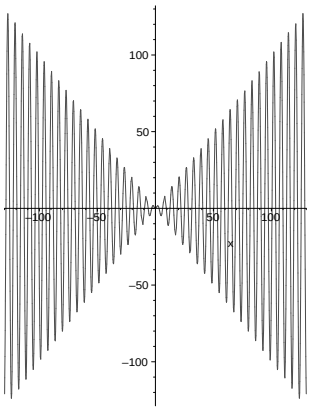
(Helyhiány miatt csak ritkítva és kerekítve írjuk ki az eredményeket.)

Ebből azt látjuk, hogy a sorozat valószínűleg 2-höz tart, és hogy $n = 12$ körül a gép által használt kerekítés már pontosan 2-t ad. Azt ne higgyük, hogy onnantól kezdve $a_n = 2$. Az sem igaz, hogy mindegyik gép vagy program ugyanúgy kerekít.

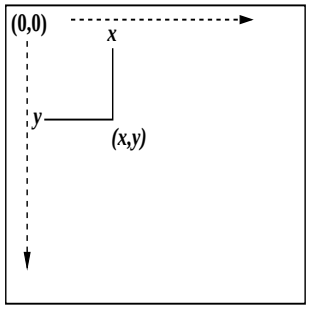
Ha más rekurzióra akarjuk alkalmazni ezt a programot, akkor két helyen kell változtatnunk: (r)-ben a rekurzión és (i)-ben az induló értékeken.

19.3.3. Függvények ábrázolása

Hosszabb előkészítés helyett a közepére ugrunk: egy függvényábrázoló programmal az $x \sin x$-et rajzoljuk ki. A képernyőre írhatunk karaktereket (karakteres képernyő) illetve kirakhatunk képpontokat egy 640×480 pontból álló téglalapra.



19.2(a) ábra: Egy bonyolult fv.



19.2(b) ábra: A lefordított koordinátarendszer

3. Program **(Függvényábrázolás).**

```
SCREEN 12                                  Grafikus képernyo
FOR x = - 300 TO 300 STEP 0.1                   ciklus eleje   (e)
y=5*i*SIN(x/10))                          A fuggveny kiszamolasa   (f)
PSET (x + 300, 200 - y)                    A keppont kirajzolasa   (p)
NEXT x                                             ciklus vege   (v)
```

Magyarázat: A SCREEN 12 grafikus képernyőre vált: azon tudunk rajzolni. Használatakor a képernyő egy pontrács, egy fejre állított koordináta-rendszerrel: a bal felső sarok a $(0,0)$, a jobb alsó pedig a $(639,479)$. A PSET (X,Y) a képernyő (X,Y) pontjába rajzol egy pontot. Az (f)-beli .5∗i∗SIN(i/10) már a skálázást is tartalmazza. Az (e) és (v) sorok adják meg a ciklust, amelyik a közrefogott (f) és (p) sort kb. 6000-szer hajtja végre: Minden az (e) által megadott x-re (f) kiszámolja a függvényértéket, (p) pedig kiteszi a megfelelő pontot. Néha az ilyen alkalmazásoknál éppen a helyes skálázás megtalálása a legnehezebb.

1. Feladat. Ábrázoljuk a fenti program (f) sorának megváltoztatásával az
   (a) $f(x)=\sqrt{\sin x}$ függvényt,
   (b) $f(x)=\cos\sqrt{x}$ függvényt,
   (c) $f(x)=\sin x+\sin 2x+\sin 3x$ függvényt.

19.4. Mit tud a QBasic?

$a_n=\left(1+\frac{1}{n}\right)^n$ **határértéke.**

Nem várhatjuk el programjainktól, hogy egy mélyen elméleti tétel bizonyításában segítsenek, de pl. az [48] 5.4. Tétel és környéke kiválóan illusztrálható ilyen programokkal. Most az a feladatunk, hogy az ott megadott sorozatokat vizsgáljuk meg programjainkkal. Erre az


(48)
Miki: Hibas referencia?

1. programot használhatnánk. Előbb azonban csiszoljuk egy kicsit. Hogyan tehetjük az 1. programot kicsit kényelmesebbé? Például úgy, hogy ciklusba szervezzük:

4. Program **(Sorozat határértéke II.).**

```
REM ciklus                                                    (r)
CLS                                                           (c)
FOR n = 1 TO 100                                              (e)
a = (1 + (1/n))^n          (A változtatható fv.)              (f)
PRINT a,                                                      (p)
NEXT n                                                        (v)
```

←— **Itt tartok** X

Magyarázat: A változatosság kedvéért itt az $\left(1+\frac{1}{n}\right)^n$-t írtuk be. Ennek hatására a gép kiírja az első 100 tagot. Ez itt megengedi, hogy megsejtsük a határértéket, de lassan konvergáló sorozatoknál esetleg teljesen félrevezethet. Az eredmény (kerekítve):[81]

$\left(1+\frac{1}{n}\right)^n :=$

1	2	3	4	5	8	100	1000	10000
2	2.25	2.37	2.44	2.49	2.57	2.7048	2.71692	2.71815

Itt is **for-next** ciklust használunk A (e)-(v) sor-pár azt intézi el, hogy a közöttük levő sorokat a program végrehajtsa $n = 1, \ldots, 100$-ra. A 2. sor egy Clear Screen, azaz képernyőtörlés, kellemesebbé teszi a program futtatását, de nem funkcionális. Az 1. sor egy ,,remark" = ,,megjegyzés", a program futását nem befolyásolja, de segít a program áttekintésében.

Programjaink szerkezete. *Programjaink egyszerű szerkezetűek: 3–5 részből állnak:*
(a) bekérik az adatot, *(INPUT)*
(b) számolnak és/vagy rajzolnak
(c) kiírjak az eredményt. [82] *(OUTPUT)*
A program ezen egységeit megelőzhetik emlékeztetők, ill. a Pascal *programoknál a fejléc és a változó-deklarációk.*

A hosszabb vagy sok ember által használt programoknál **életfontosságú az áttekinthetőség:** *a program, az input és az output áttekinthetősége.*

A PRINT-ről azt kell tudni, hogy utána változókat, szövegeket (stringeket) és matematikai kifejezéseket írhatunk, ezeket ; -vel és , -vel elválasztva egymástól. Ha a sor végére semmit nem írunk, a PRINT sort emel. A ; és , szabályozza, hogy egy kiírás után a következő PRINT hova írjon: ; után csak minimálisat megy tovább, , egy tabulátornyit ugrik. A részleteket itt átugorjuk. Ha szöveget akarunk kiírni, azt idézőjel közé tesszük.

Az **INPUT "sebesseg"; a** a **PRINT "sebesseg"; : INPUT a** rövidítése. Mindkettő **feliratozott INPUT**: bekér egy adatot, de előbb kommentálja.

[81] $n > 100$-ra (e) sor átírandó, és érdemes felcserélni a két utolsó sort. Miért?
[82] Az eredmény lehet maga a rajz is.

2. Feladat. (a) A 19.4. programba írjuk be a következő sorozatokat:

$$\left(1-\frac{1}{n}\right)^n, \qquad \left(1+\frac{1}{n}\right)^{n^2}, \qquad \left(1+\frac{1}{n^2}\right)^n,$$

és sejtsük meg határértékeiket a géppel. Ez kicsit nehezebb az első esetben. Igazoljuk sejtésünket gép nélkül (is).

(b) Vizsgáljuk meg a 4 fejezet, 4.1 Példa képlettel megadott sorozatait!

Az alábbiakban konvergenciasebességet fogunk becsülni. Egy $a_n \to a$, sorozat konvergenciájának sebessége lehet gyorsabb vagy lassabb (ami azt jelenti, hogy a $h_n = a_n - a$ sorozat gyorsabban vagy lassabban tart 0-hoz). Mi arra vagyunk kíváncsiak, hogy a h_n különbség, mondjuk $\frac{c}{n}$-nel becsülhető-e vagy $\frac{c}{\sqrt{n}}$-nel, esetleg csak a sokkal nagyobb $\frac{1}{\log n}$-nel?[83] A konvergenciasebesség becslésére itt azt a módszert használjuk, hogy a különbséget különböző nagyságú sorozatokkal beszorozva megvizsgáljuk, hogy a szorzat mikor korlátos és mikor nem, mikor tart 0-hoz. Minél jobb becslést szeretnénk találni.

Az $\sqrt[n]{n}$ sorozat konvergenciasebességét vizsgáljuk; első lépésként a $b_n = n \cdot h_n = n(\sqrt[n]{n} - 1)$ sorozatról szeretnénk megtudni, tart-e 0-hoz.

5. Program **(Konvergenciasebesség).**

```
     REM n-edik gyok n, ciklus
     CLS
10   INPUT "n = "; n
     IF n < .01 THEN END                    (Megállás feltétele)
     a = n*(n^(1/n) - 1)
     PRINT n,a
     GOTO 10                                (Visszatérés 10-re)
```

Magyarázat: Ez a program tipikusan olyan, amilyet az ember csak saját magának készít, hogy valamit gyorsan kiszámítson. Nem érdemes túlcicomázni.[84] Arra való, hogy egymás után az adott sorozat akárhány tagját kiszámítsuk. A számolás elvégeztével a program visszaugrik a **10**-es „címkére": végtelen ciklusban dolgozik. Enter-rel ugorhatunk ki belőle, mert az Enter az n-et 0-ra állítja. A program leállítása egy tetszőleges $n \leq 0$-val történik, de a program pozitív valós számokra is működik. Vigyázat, ha leállító feltételnek $n \leq 0$-t írtunk volna, akkor nem tudnánk, hogyan viselkedik a program $n = 10^{-100}$ körül: leáll-e vagy sem.

Ha egy Basic programot hamarabb akarunk leállítani, mint ahogyan magától leállna, használjuk a Ctrl + Break-et[85].

Mit kapunk ezzel a programmal? (Helyhiány miatt csak ritkítva és kerekítve írjuk ki az eredményeket.)

[83] Vigyázat, mi a $\log x$-et a természetes alapú logaritmusra használjuk, de néha rákényszerülünk az $\ln x$ (=ln(x)) használatára is, pl. a Maple-ben.

[84] Itt már az első két sor is cicomázásnak számíthat, elhagyható.

[85] Ez persze nem működik, ha a gépünk úgy van beállítva, hogy ne működjék.

$n(\sqrt[n]{n}-1)$:

$n=$	2	3	4	6	100	1000	10^4	10^6
$n\cdot h_n=$	0.83	1.33	1.66	2.09	4.71	6.93	9.21	13.8

Ebből megsejthetjük, hogy az $\sqrt[n]{n}\to 1$-ben a hiba nagyobb nagyságrendű, mint $1/n$. Kipróbálhatjuk, hogy kisebb, mint $1/\sqrt{n}$. Ha viszont $\frac{\log n}{n}$ hibával, azaz $h_n=\frac{n}{\log n}(\sqrt[n]{n}-1)$-nel próbálkozunk, a következőt kapjuk:

$\frac{n}{\log n}(\sqrt[n]{n}-1)$:

$n=4$	5	6	100	1000	10^5	10^6
1.195	1.18	1.17	1.023	1.0035	1.000058	1.000007

Ebből azt sejtjük, hogy $\sqrt[n]{n}-1\sim\frac{\log n}{n}$. (Ez a példa annyiból csak ,,iskolapélda", hogy nem túl nehezen be is bizonyíthatjuk ezt a sejtésünket. Lásd 10.32.(e) feladat.)

3. Feladat. Szeretnénk vizsgálni az $a_n=\left(1+\frac{1}{n}\right)^n$ sorozat konvergenciasebességét. Módosítsuk az előző programok valamelyikét úgy, hogy segítsen ezt megsejteni.

4. Feladat. Nézzük meg programmal, mi a határértéke és a konvergenciasebessége az $a_n=\sqrt[n]{2}$-nek, illetve az $a_n=\sqrt{n+1}-\sqrt{n-1}$-nek.

5. Feladat. Alakítsuk át a fenti programot úgy, hogy ε legyen az INPUTja, és ehhez keressen küszöbindexet. (Vigyázat, ez csak heurisztikus szinten valósítható meg. Ha a határértéket is tudjuk, és bekérhetjük, és a sorozat monoton, arra persze írható ilyen program.)

Megjegyzés 19.1 (Nehéz-e egy programot megírni?) *Ezen rész elején azt mutattuk meg, hogy 6-soros programokkal már nagyon bonyolult analízisbeli kérdésekhez is kaphatunk segítséget. Egy adott programhosszúság felett azonban már a program átláthatatlanná válhat. A 417. oldalon a 19.4.5. program már jóval hosszabb, de még áttekinthető, és a bonyolultságát nem a hossza, hanem az egymásbaágyazott ciklus adja.*

19.4.1. Sorösszegzéssel kapcsolatos programok

Reciproknégyzetösszeg. Nagyon fontos és nem triviális matematikai tétel, hogy $\sum_{i=1}^{n}\frac{1}{i^2}\to\frac{\pi^2}{6}$, ha $n\to\infty$. Ez már szerepelt a 6.11 Példában, illetve bebizonyítottuk a 9.4.36. feladatban. [49] Most ezt ,,vizsgáljuk meg" számítógéppel.


(49)
Miki: Kis F?

6. Program (**Reciprok-négyzetösszeg**).

```
PRINT "Reciproknegyzetosszeg"
INPUT "n = "; n                    (i)
FOR i = 1 TO n
s = s + (1/i)^2                    (s)
NEXT i
PRINT s                            (p)
PRINT SQR(6*s);                    (q)
```

Magyarázat: Mivel sem az első sor, sem az utolsó nem szükséges igazán a programhoz, a limesz (sorösszeg) keresésére tulajdonképpen egy 5-soros programot használunk. A Basic futtatáskor lenullázza a változókat. Ha nem tenné, akkor a ciklus elé be kellene írnunk, hogy $\mathbf{s = 0}$. (i) bekéri, hogy meddig adjuk össze, (s) a ciklusban az összeget képzi, az s változóba rakva be a mindenkori összegértéket. A (p) sor kiírja az összeget, amiről érezhető, hogy konvergál, a (q) sorban kiírjuk $\sqrt{6s}$-t, arról sejthető, hogy π-hez tart. Ebből érezhető, hogy $s \to \frac{\pi^2}{6}$.[86]

6. Feladat. A $\sum_{k=1}^{n} \frac{1}{k} - \log n$ sorozat konvergens: az Euler-konstanshoz tart (lásd a 9.3:17. feladatot). Írjunk programot ennek illusztrálására: írjuk kicsit át a fenti programot. [50]


(50)
Miki: Hogyan szamozzam a feladatokat?

A görbe alatti terület. A 6 és a 218. oldalakon a parabola alatti területrészt számítottuk ki. A következő program ezt illusztrálja. [51]


(51)
Miki: Program formatum?

7. Program **(A parabola alatti terület).**

```
     REM teruletszamitas
(i)  INPUT " a = "; a : INPUT " b = "; b : INPUT " n = "; n
(e)  PRINT "a vizsgalt intervallum : ["; a; ","; b; "]"
(h)  lep = (b - a)/n : s = 0
     PRINT "felosztasszam = "; n, "lepeshossz = ", lep
(c)  FOR i = 1 TO n                                    (i-ciklus eleje)
     x = a + i*lep
(f)  y = x*x
     s = s + y
(v)  NEXT i                                            (i-ciklus vége)
     PRINT "A terulet kozelito erteke:  ", s*(b - a)/n
```

Magyarázat: (i)-ben bekérjük az intervallum végpontjait, továbbá, hogy hány felosztó pontot használunk. (e) pusztán ellenőrzésképpen kiírja ezeket. (h) kiszámítja a felosztással kapott kis intervallumok hosszát. A (c)–(v) ciklus végzi az igazi munkát, az s-ben összeadogatja a kis téglalapok területeit: pontosabban a magasságukat, amelyet az utolsó sorban megszoroz a téglalapok közös oldalhosszával, ,,lep"-pel.

Ez a program is **keretprogram:** akármilyen függvényre alkalmazható. A programban csak egyetlen sor tükrözi, hogy az $y = x^2$ függvénnyel van dolgunk, az (f) sor. Ha ezt a sort más függvényre cseréljük, annak a grafikonja alatti (előjeles) területet kapjuk meg.

Érdemes megjegyezni, hogy az ún. trapézösszegek lényegében ugyanennyi munkával sokkal jobban közelítenek. Ebben a kis görbeszakaszok alatti területet trapézokkal közelítjük. Matematikailag ez csak annyi változást okoz, hogy az első és utolsó pontokban $f(x)$-et csak $\frac{1}{2}$ súllyal számítjuk be.

[86] Nem gondoljuk, hogy ez ki is található; egy bizonyításvázlatot adunk erre a 6.36 feladatban.

7. Feladat. Módosítsuk a fenti programot úgy, hogy trapéz összeget számítson. Próbáljuk ki az x^2, ill. a $\sin x$ függvényeken, ez mennyivel jobb a téglányösszegnél. [52]


(52)
Miki: Helyesira?s

19.4.2. Rekurziók: A Newton-algoritmus

A következő program az $a_{n+1} = \frac{1}{2}\left(a_n + \frac{a}{a_n}\right)$ rekurzióval definiált sorozat határértékét vizsgálja. Ez az ún. Newton-féle gyökkereső algoritmus legegyszerűbb esete, a négyzetgyökvonásra alkalmazva. (Lásd az 5.1.28. feladatot.) A közelítés szemléletes magyarázata az, hogy ha egy adott a-hoz $b = a_n$ jól közelíti $\sqrt{a}$-t felülről, akkor a/b is jól fogja közelíteni, de alulról, a számtani közepük pedig még sokkal közelebb lesz $\sqrt{a}$-hoz. Sőt, mivel $b - \sqrt{a}$ és $\sqrt{a} - \frac{a}{b}$ közel egyenlőek, a számtani közép sokkal-sokkal jobban közelíti $\sqrt{a}$-t.

8. Program **(Newton-gyökvonás).**

```
REM newton
DEFDBL A-B                                  Dupla pontosság | (d)
INPUT "a = "; a
INPUT "Hany iteraciot:  "; n
b = 1
FOR i = 1 TO n
b = (b+(a/b))/2                             (a rekurzió)    | (r)
PRINT i, b
NEXT i
```

Magyarázat: A rekurzió (r)-be van beírva. A (d) sor dupla pontosságúvá teszi a-t és b-t. Enélkül nem igazán figyelhetnénk meg a konvergencia hibáját, mert néhány lépés után eltűnne a kerekítésben. Ez kiderül, ha a programot alkalmazzuk valamilyen négyzetszámra, mondjuk 4-re, a (d)-beli **DEFDBL** elhagyásával.

Mit figyelhetünk meg? Ha pl. a programot 10000-re alkalmazzuk, akkor eleinte a hiba feleződik, majd amikor a hiba már pl. $1/2$ alá megy, akkor lényegében minden hiba az előző hiba négyzetével becsülhető:

Ha $a \geq 1$ és $a_n = \sqrt{a} + h_n$, akkor

$$a_{n+1} - \sqrt{a} = \frac{1}{2}\left(a_n + \frac{a}{a_n}\right) - \sqrt{a} = \frac{1}{2}\left(\sqrt{a} + h_n + \frac{a}{\sqrt{a} + h_n}\right) - \sqrt{a} =$$
$$= \frac{1}{2}\left(\sqrt{a} + h_n + \frac{a - h_n^2}{\sqrt{a} + h_n} + \frac{h_n^2}{\sqrt{a} + h_n}\right) - \sqrt{a} = \frac{1}{2}\frac{h_n^2}{\sqrt{a} + h_n} < \frac{h_n^2}{2}.$$

Természetesen ez pl. $h_n = 2$ esetében nem sokat ér, de ha egyszer $h_n < 1$, onnan az a_n sorozat gyorsan konvergál $\sqrt{a}$-hoz.

19.4.3. Rekurzió π közelítésére

A π ,,kiszámolása" izgalmas kérdés: A matematika egyik legfontosabb konstansáról van szó. A π a matematikusok számára nem csak az egységkör félkerülete: egy fontos konstans a analízisben, számelméletben, valószínűségszámításban, komplex függvénytanban, stb. Mint láttuk, a π a Stirling-formulában is felbukkan. (L. (19) képlet, 83 oldal.)

Az alábbi elemi geometriai feladat megoldására van szükségünk.

8. Feladat. Bizonyítsuk be, hogy ha a_n az egységkörbe írható szabályos n-szög oldalhossza, és $m_n = a_n^2/4$, akkor

$$m_{2n} = \frac{1}{2}(1 - \sqrt{1 - m_n}). \tag{238}$$

Induljunk ki az egységkörbe írt szabályos 6-szögből. Erre $m_6 = 1/4$. (238) alapján $m_{12}, m_{24}, m_{48}, m_{96}, \ldots$ rekurzíve könnyen számolható, és $\pi = (1/2) \cdot \lim n \cdot a_n$. Ezen alapul az alábbi, ,,Archimedesz-i" program:

9. Program **(Szabályos sokszögek a körben).**

```
     REM pi kozelitese
(d)  DEFDBL S, M                        (Dupla pontosság)
     INPUT "Hany duplazas "; z
(i)  n = 6 : m = .25
     FOR i = 1 TO z
(r)  m = (1 - SQR(1 - m))/2                  (a rekurzió)
(k)  n = 2*n
(u)  s = n*SQR(m)                       (A közelítő kerület)
     PRINT i, s
     NEXT i
```

Magyarázat: (d) azt kéri, hogy a kerületet dupla pontossággal számolja a gép: DBL a double = dupla rövidítése. Az (i) sor beállítja a rekurzió kezdőértékét: szabályos hatszögre az oldalhossz 1, így $m_6 = \frac{1}{4}$. (r)-ben számítjuk ki a $2n$ oldalszámú szabályos sokszög oldalhosszának megfelelő m_{2n}-et, (k)-ban kétszerezzük az oldalszámot, (u)-ban számoljuk az új sokszög kerületét.

$6 \cdot 2^{10}$	3.141593	$6 \cdot 2^{27}$	3.140275	$6 \cdot 2^{31}$	3.000000
$6 \cdot 2^{20}$	3.141593	$6 \cdot 2^{29}$	3.137475	$6 \cdot 2^{32}$	0
$6 \cdot 2^{23}$	3.141587	$6 \cdot 2^{30}$	3.181981	Baj van!	

A program $z = 20$-szal jó eredményt ad, $z = 30$-cal túlmegy a 3.18-on, ami csak rossz lehet. $z = 40$-re 0-t: teljesen rossz eredményt ad! Már a 20. és 23. lépés között is rossz, hiszen az eredményeknek monoton növekedniük kellene. (Ha korábbi Basic-et használunk, hamarabb jutunk rossz eredményre.)

A baj abból származik, hogy a kerületet, mint az oldalszámszor oldalhosszat, $\infty \cdot 0$ típusú határértékként közelítjük. Ez könnyen kijavítható, ha felhasználjuk, hogy

$$1 - \sqrt{1 - m} = \frac{m}{1 + \sqrt{1 - m}}, \tag{239}$$

és az itteni jobboldalt írjuk be a program (r) sorába. A részleteket átugorjuk, a gyöktelenítés éppen a numerikusan instabil kis különbségektől való megszabadulást jelenti.

19.4.4. Stirling-formula

A Stirling-formuláról volt már szó (a 83, ill. 415. oldalon). Vannak pontosabb és durvább formái. Mi számítógéppel a 13.14. tételben kimondott formáját illusztráljuk, amely szerint

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n. \tag{240}$$

10. Program **(Stirling Formula).**

```
REM === Stirling Formula ===
DEFDBL K, P                                                    (d)
CLS : INPUT "n = "; n
REM === A pontos ertek ===
pontos = 1                                                     (p)
FOR i = 1 TO n : pontos=pontos*i : NEXT i   (Ez számolja a n!-t)
REM === A kozelites ===
e = 2.71828182845#                                             (e)
kozel = (n/e)^n
kozel = kozel*SQR(2*n*3.141592653#)              (n!-közelítése)
PRINT n; " | "; pontos; " | "; kozel,
PRINT " arany:  "; pontos/kozel
```

Magyarázat: Itt olyan nagy számokkal is dolgozunk néha, hogy nagyon óvatosnak kell lennünk. Az egyik óvatossági lépésünk, hogy (d)-ben dupla pontosságot állítunk be. A (p) és az utána következő ciklus kiszámolja $n!$ pontos értékét. Utána (240) alapján kiszámoljuk a közelítő értéket, majd kinyomtatjuk a két értéket és a hányadosukat. A $3.14\ldots$ és a $2.71\ldots$ utáni # jelet a gép tette oda, jelezve, hogy e két számot dupla pontosságúnak veszi.

Próbáljuk ki a fenti programot. Ha $\alpha(n)$-nel jelöljük a hányadost, $\alpha(5) = 1.016$, $\alpha(7) = 1.0119,\ldots$, $\alpha(20) = 1.00417,\ldots$. A Stirling-formula tehát már kis értékekre is jól közelít, és később tovább javul. (Vigyázat, ez nem bizonyítás!)

9. Feladat. A Stirling formula legközvetlenebb szokásos finomítása

$$n! \approx \left(1 + \frac{1}{12n}\right)\left(\frac{n}{e}\right)^n \cdot \sqrt{2\pi n}.$$

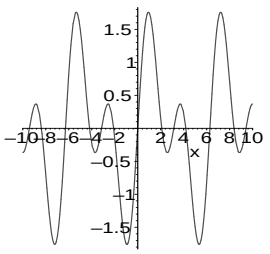
Próbáljuk ki ezt a formulát is, a 19.4.4. Program kis megváltoztatásával.[87]

[87] Itt ≈-ot nem definiáltuk. Mégis, fogalmazzuk meg, mit tapasztalunk.

19.4.5. Függvények ábrázolása II.

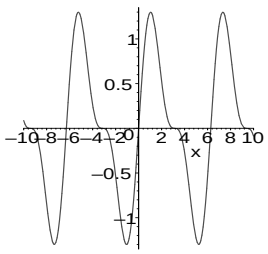
Az analízisben segítségünkre lehet egy jó ábra egy fogalom megértésében vagy bizonyos esetekben az igazság megsejtésében. Ehhez gyakran segít a megfelelő program.

Trigonometrikus polinomok. A matematikában nagyon fontosak és érdekesek a trigonometrikus polinomok. Ezek a $\sum_{n=1}^{N}(a_n \sin nx + b_n \cos nx)$ alakú függvények. Az alábbiakban néhány ilyet mutatunk be.



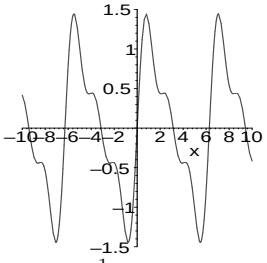
$\sin x + \sin 2x$

19.3(a) ábra



$\sin x + \frac{1}{2}\sin 2x$

19.3(b) ábra



$\sin x + \frac{1}{2}\sin 2x + \frac{1}{3}\sin 3x$

19.3(c) ábra

A 3(a) ábrán az $f(x) = \sin x + \sin 2x$, a 3(b)-n az $f(x) = \sin x + \frac{1}{2}\sin 2x$, a 3(c)-n pedig az $f(x) = \sin x + \frac{1}{2}\sin 2x + \frac{1}{3}\sin 3x$ látható. Kérdezhető, hogy az

$$f_n(x) = \sum_{k=1}^{n} \frac{1}{k} \sin kx \tag{241}$$

függvények sorozata konvergál-e valamilyen értelemben egy függvényhez. (Ebben a fejezetben a **pontonkénti konvergenciára** szorítkozunk. Akkor mondjuk, hogy az f_n függvények sorozata az A halmazon az f függvényhez konvergál, ha $f_n(x) \to f(x)$ minden $x \in A$-ra, amibe az is beleértendő, hogy minden $x \in A$-ra $f(x)$ véges és $n > n_0(x)$-re f_n értelmezve van.)

10. Feladat. Próbáljuk megsejteni, hogy mi a határértéke a (241) függvénysorozatnak. (Kifejezhető a ,,törtrész" függvénnyel!)

Más szóval az a kérdés, hogy mi mondható a (241)-ben megadott függvénysorozatról? Az alábbiakban tehát egy olyan programot írunk, amelyik a (241)-ben megadott $f_n(x)$-et rajzolja ki a képernyőre: célja, hogy kísérletezzünk, szemlélgessük a közelítés sebességét. A program meglehetősen ,,fapados", a kényelmesebbé tételére szolgálnak az utána következő feladatok.

[53]

(53)

Miki: Be nem valtott igeret

[54]

Miki: Kati, ez meg rendbehozando

11. Program **(Fűrészfog).**

```
SCREEN 12                                                                    (s)
INPUT " n= "; n : INPUT " magassag= "; magas      : INPUT " b= "; b          (i)
CLS                                          (Letörli a képernyőt)           (c)
FOR i = 0 TO 639                               (i-Ciklus kezdete)
yregi = y
x = i*b/639 : y = 0
FOR m = 1 TO n : y = y + magas*SIN(m*x)/m                  : NEXT m          (F)
PSET (i, 240) : PSET (i, y + 240)                                            (p)
IF i > 0 THEN LINE(i - 1, yregi + 240)                  -(i,y + 240)         (v)
NEXT i                                                 (i-ciklus vége)
PRINT "n = "; n, "magas = "; magas,                       "b = "; b,
REM Jo adatok:  5, 20, 30, vagy 50, 80,                      50, ...
```

Magyarázat: Az (i) sorban a program három paraméterét kérjük be. Szabad egy sorba több utasítást írni, de [:]-tal kell azokat elválasztani. Itt éppen azt kértük be, hogy hány tagú legyen a szinuszpolinom, hogyan skálázzuk y irányban a görbét (avégett, hogy ne legyen túl lapos, de ki se fusson a képernyőről), és hogy milyen $[0, b]$ intervallumon dolgozzunk.

Itt három grafikus utasítást használunk: a grafikus képernyőre váltást (s)-ben, a pontkirakást a (p)-ben, a simább görbe érdekében viszont yregi-be mentjük el y értékét, majd (v)-ben az új (i,y) pontot összekötjük a régivel: a **LINE(i,y)–(i − 1,yregi)** a két pont közé húz egy egyenes szakaszt.

(F) számolja ki a trigonometrikus polinomot. A (c) sor letörli a képernyőt. (Elhagyható?) Ha egy sorban IF-THEN-t használunk, akkor az IF feltétel teljesülése esetén (l. a (v) sor) az adott sorban minden utána következő utasítást végrehajt. (p) első fele kirajzolja az x tengely i-edik pontját, a második fele pedig a felette levő pontot. A $+240$ a görbét függőlegesen középre tolja.

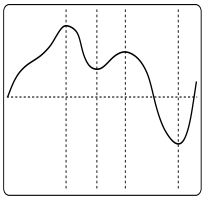
Ha a (v) sort elhagyjuk, a program ugyanúgy fog működni, az előző sor második fele kirakja a pontokat. Amit kapunk, az néha kicsit szaggatott lesz. Ez a sor egyenes szakaszokkal köti össze a konzekutív pontokat, így a kapott kép – aránylag kevesebb pont esetén – sokkal szebb lesz. Viszont $i = 0$-ra ez nem alkalmazható: nincs megelőző pont.

Az utolsó előtti sor felesleges: előfordulhat, hogy lefuttatjuk a programot, megtetszik a kapott kép, de hirtelen nem emlékszünk, mik voltak az input adatok. Ez a sor kiírja azokat. Az utolsó sor feljegyzés magunk számára, hogy milyen változókkal sikerült kellemes képet kapnunk.

A kapott függvények pontonként a $\{x\}$ (törtrész) függvény egy lineáris transzformáltjához tartanak. (Melyikhez?)

Az alábbi feladatokban implicite feltesszük, hogy a függvényeink nem túl vadak: ismerve azon 200-2000 pontban az értékeiket, ahol megvizsgáljuk őket, a köztes intervallumokban jó közelítéssel lineárisoknak képzelhetjük őket.

11. Feladat. Változtassuk meg a 19.4.5. programot úgy, hogy



19.4. ábra:

(a) előbb kiszámolja a függvény maximumát és minimumát, majd – egy második menetben, a maximum és minimum ismeretében – automatikusan olyan skálázást használjon, amelyikben a függvény görbéje elfér a képernyőn, de egyben azt kellemesen ki is tölti.
(b) Mindig tegye el az utolsó előtti és az azelőtti függvényértékeket, és ha az utolsó előtti nagyobb az előtte és utána következőnél, akkor ott húzzon egy függőleges vonalat (jelezve, hogy ott valószínűleg maximumhely van).

12. Feladat. **(Függvényábrázolás diszkusszióval)** Változtassuk meg az előző programot úgy, hogy

(c) ahol a függvény monoton növekszik, ott pirossal rajzolja ki a függvény pontjait, (**color(4)**), ahol a függvény monoton csökken, ott kékkel (**color(1)**).

(d) a (c) helyett jelezzük ki a konvex és konkáv szakaszokat. (A konvexekre h lépésköz esetén $f(x) - f(x-h) > f(x-h) - f(x-2h)$ jellemző.

←— **Itt tartok** B

19.4.6. Görbeseregek ábrázolása

13. Feladat. Az e^x függényt az tünteti ki az a^x alakú függvények között, hogy a 0-ban 1 a meredeksége, $f'(0) = 1$. Ennek vizsgálatára írjunk egy programot, amelyik $f(x, a)$ alakú, a-val paraméterezett görbeseregeket rajzol ki. (Itt $f(x, a) = a^x$.)

Megoldás: [55]

12. Program (**Görbeseregek).**

(55)

Miki: Kati: itt a betutipustol függ a ,,minusz": - vagy –

```
(1)  SCREEN 12
(2)  ax = 100 : ay = 100 : lep = .05 : e = 2.718281828#
(3)  x = 0 : FOR y = -1 TO 2 STEP .03 : GOSUB rajzol : NEXT y
(4)  y = 0 : FOR x = -2 TO 4 STEP .03 : GOSUB rajzol : NEXT x
(5)  LINE (200 - 200, 200 + 200)-(200 + 200, 200 - 200)
(6)  FOR a = e - 2 TO 4 STEP .25
(7)  FOR x = -2 TO 4 STEP lep
(8)  IF ABS(a - e) < .1 THEN szin = 4 ELSE szin = 3
(9)  yregi = y : xregi = x - lep :
(10) y = a^x :
(11) GOSUB rajzol
(12) NEXT x
(13) INPUT w
(14) NEXT a
(15) END
(16) rajzol:                                  (A rajzoló szubrutin belépési pontja)
(17) IF x < - 1.9 THEN RETURN
(18) LINE (200+ax*xregi, 300-ay*yregi)-(200+ax*x, 300-ay*y), szin
(19) RETURN                                   (A szubrutin visszatérési pontja)
```

Magyarázat: Ebben a programban a kirajzolást egy ,,rajzol" nevű szubrutinnal oldottuk meg. Három különböző helyről is meghívjuk: a (3), (4) és a (11) programsorokból. A szubrutin a LINE paranccsal egy kis vonalkát húz az előző pont és az új pont között. Ha csak egy pontot tennénk ki, a kapott görbe szaggatottabb lenne. A RETURN hatására a program mindig az "ideküldő" gosub utáni parancsra tér vissza.

Rajzoláskor gondot szokott okozni, hogy az ábrák kifutnak a képernyőről vagy pedig nagyon kicsire (a függvények nagyon laposra) sikerülnek. Ezzel többnyire el kell játszanunk. A szubrutinos formának itt elsősorban az az előnye, hogy a beskálázás így könnyebb.[88]

Az (1)-ben térünk át a grafikus képernyőre. A SCREEN 12-es képernyőmód biztosítja, hogy színesen rajzolhassunk. (2)-ben állítjuk be a paramétereket: ,,ax", ,,ay" az x, illetve y irányú nagyítás, ,,lep" a lépésköz, e a természetes logaritmus alapja. ,,szin" adja a színt. Itt csak annyi kell nekünk, hogy amikor az a^x-ben az a e-hez közel jut, akkor más színnel rajzoljunk ki.

(3) és (4) a koordinátatengelyeket rajzolja ki. Mit csinál (5)? [56]


(56)
Miki: Ezt leellenorizni?

A program lelke a (9)–(11) rész: ha más függvényre vagyunk kíváncsiak, csak a (10)-et kell megváltoztatnunk és a skálázást. (9)-ben ,,adatot mentünk": eltesszük az előző értéket, hogy (18)-ban használhassuk.

Két egymásba ágyazott ciklust használunk. A görbesereg paraméterét, a-t a (6) és a (14) sor szervezi ciklusba. A (13) sor minden görbe kirajzolása után megállítja a programot azzal, hogy bekéri w értékét. (Erre az értékre persze nincs szükségünk: (13)-at csak megállításra használjuk.) Elég csak az ENTER-t lenyomnunk.

[88] Nagyobb programoknál elengedhetetlen a programok kisebb egységekre, mondjuk, szubrutinokra bontása. De itt mi nem foglalkozunk nagyobb programokkal.

A (15)-beli END azért kell, hogy a szubrutinba csak a GOSUB-on keresztül érkezhessünk be. Enélkül az utolsó a után is bemegy a szubrutinba, majd hibát jelez.

(8) intézi el, hogy e^x más színnel jelenjen meg, mint a többi hatványgörbe.

19.5. Rövid kirándulás a PASCAL-ba.

Nehéz-e a PASCAL?

Nem, a PASCAL egyáltalán nem nehéz. Itt nem fogjuk nagyon elmagyarázni, csak megadunk néhány programot BASIC-ben és Pascalban is, egyaránt. A Pascal fő előnye az, hogy sokkal fegyelmezettebb nyelv, mint a BASIC, de számunkra ez az előnye nem lesz érzékelhető. Elsősorban akkor ajánljuk, ha a gépünkön PASCAL van telepítve, vagy ha PASCAL-t tudunk könnyen telepiteni.

Megjegyzések a PASCAL-hoz

Itt abból indulunk ki, hogy valaki már ismeri a BASIC-et, de szeretne áttérni PASCAL-ra. Ilyenkor az alábbi programokat megszemlélve az áttérés nagyon könnyű. Néhány különbséget azonban megfogalmazunk az alábbiakban.

Tekintsük a következő PASCAL programot, amelyik egy derékszögű háromszög átfogóját számolja ki.

13. PROGRAM (**Pithagoras, PASCAL**).

```
program Pithagoras;                                    fejléc
var a,b,c:real;                                változók deklarációja
begin
  readln(a);                                   1.  befogó beolvasása
  readln(b);                                   2.  befogó beolvasása
     c:=sqrt(a*a+b*b);                            átfogo kiszámolása
  writeln(c:3:5);                 Az eredmeny kiiratása 5 tizedesre
end.
```

1. A PASCAL program egy fejléccel kezdődik, ahol megmondjuk a program nevét, majd a programban szereplő változókat, és azok típusait. Fentebb erre szolgál a első két sor.
2. Maga a program, (tehát a fejléc és a változók ,,deklarációja" utáni rész) egy "begin" es egy "end." között van.
3. A BASIC INPUT-ja helyett itt readln-t használunk, ahol az "ln" arra való, hogy a beolvasott adatot Enterrel kelljen befejeznünk. Ha pl. csak egy "ch" karaktert szeretnénk beolvasni, akkor nem volna szükségünk az adat végének jelzésére, igy

$$ch := readkey; \qquad \text{vagy} \qquad read(ch);$$

utasításokat használnánk.[89]

4. Ha egy számot kiíratunk, megadhatjuk, hány jegye legyen a tizedespont elött és után: a fentiekben a writeln(c:3:5); azt jelenti, hogy az egész részt 3 jegyre, a tizedespont utáni részt 5 tizedesjegyig írjuk ki. [90]
5. A BASIC és PASCAL programjaink közti egyik különbség, hogy a BASIC nem igényel fejlécet, változó-deklarálást, ill. az aktuális programrész elejének és végének külön kijelzését.

←— Itt tartok Y

Egyelőre ennyi elegendő lesz, sok "információ" található a PASCAL-ról az INTERNET-en, és a honlapomon (237).

19.5.1. A fenti program PASCAL átírása

14. PROGRAM (**Konvergencia sebesség, PASCAL verzió).**

cimke	A programsor	a magyarázata
	program konvergseb;	fejléc
	var n:integer	változók deklaralása
	a:real;	
	begin	
	clrscr;	képernyő törlése
	write('n= '); realn(n);	Input beolvasása
	a:=n*(n^(1/n)-1);	(???)
	writeln(n:4,a:4:6);	6 tizedesre való kiírás
	end.	

Ebben a programban a (???)-vel megjelölt sor ROSSZ: az odaírt képletet átvettük a BASIC-ből. Valójában a problémát az $\ln(x)$ és az $exp(x) = e^x$ segítségével kell megoldanuk. Hogyan?

19.6. MAPLE: Első lépések

Hogyan ismerünk meg egy ilyen óriási programot? Legegyszerűbb megkérni valakit, aki ért hozzá, hogy mutasson meg nekünk bizonyos egyszerűbb dolgokat, mindenekelőtt azt, hogy (A) hogyan lehet elindítani a programot, (B) kiszállni a programból, és (C) hogyan lehet elmenteni, amin dolgoztunk, (D) hogyan használatjuk a HELP-et. A MAPLE esetében ezek egyszerűek.

19.6.1. A HELP

Bármilyen, kicsit is bonyolultabb programot használunk, nagyon fontos a HELP (magyarul SÚGÓ) használatának elsajátítása. Erre főleg akkor van szükségünk, ha nincs

[89] A **readkey** használatához a 2. sorban még be kell hívnunk egy szubrutin könyvtárat, a "uses crt; " sorral.

[90] Itt az egészekre vonatkozó állítás csak annyit jelent, hogy ha a számok 1000 alatt vannak, akkor a tizedes pontok egymás alá kerülnek.

a programhoz megfelelő leírásunk. Ezért érdemes a program használata előtt a HELP-jével is megismerkednünk. A MAPLE HELP-je nagyon jó. Van egy ,,New User's Tour" (azaz, séta (a MAPLE bemutatása) újoncoknak), **57** illetve Topic Search (téma szerinti keresés). A HELP mintapéldákat is tartalmaz. [58]

Persze ezek használatához kell egy kicsi angol tudás, és ha pl. parciális törtekre szeretnénk bontani, akkor a **parcfrac** kulcsszó segít, ha Lagrange-interpolációt keressük, akkor nem a Lagrange-nál, hanem az Interpolation-nál találjuk: kell egy kis találékonyság is.

57
Sharp curve

(58)
Miki: Ezt kritizaltad?

Input-Output formátum

A gépbe valahogyan be kell vinnünk a feladatot, hogy azt megoldhassuk. A MAPLE újabb verzióiban ezt több módon is megtehetjük, és az csak beállítás kérdése, hogy melyik módot használjuk. A HELP-ben erre elegendő információt kapunk. Mi a korábbi, kevésbé szép, de egyszerűbb módszert választjuk: a MENÜ-ben odamegyünk a TOOLS → OPTIONS → DISPLAY pontokhoz, és beállítjuk, hogy ,,Input: Maple Notation", és ,,Output: 2-D Math Notation". Itt állíthatjuk be a kiíratott tizedesjegyek számát is,...

Ha a munkánkat el szeretnénk menteni, azt több formában is megtehetjük. Amikor a File legördülő menüjében a New-ra kattintunk, akkor megkérdezi a MAPLE, hogy "Worksheet" vagy "Document" módot akarunk-e. Mi azt javasoljuk, a ,,worksheet" beállítást válasszuk. Az elegánsabb beállítást ugyancsak megkapjuk a Help-ből.

Ugyanennek részletesebb leírását megtaláljuk az Internet-en, illetve a Simonovits Miklós honlapon, (lásd (237)) van egy ennél részletesebb leírás is.

Minden ilyen programnál legelőször azt kell megtudnunk, hogyan kell elindítanunk és hogyan kell kiszállnunk belőle. Bizonyos LINUX verziókban az **xmaple** parancs indítja el a MAPLE programot, (bár csinálhatunk hozzá indító ikont is), WINDOWS-ban a ikonnal szoktuk elindítani.

A MAPLE ,,munkalapokkal" dolgozik, ún. ,,Maple Worksheet"-ekkel, ezek elmenthetők, .mw kiterjesztéssel.) Ha egy munkánkat elmentjük, később újra betölthetjük[91].

Amikor elindítjuk a MAPLE-t, kapunk egy ilyen ,,untitled.mw" nevű munkalapot, arra írjuk be a parancssorokat.

(a) Ha az input formát ,,Maple Input"-ra állítottuk, (lásd 422 oldal) akkor minden beírt parancsot ; -vel vagy : -tal fejezünk be. [92]

[91] Az eredmény kimenthető (,,exportálható") LATEX-ben is. Ilyenkor a készített ábrákat POSTSCRIPT formátumban, .eps kiterjesztéssel is kimenti. A kozelit.mw munkalap LATEX kimentésekor keletkezik egy (kicsit használhatatlan) kozelit.tex file, és az ábrákat kozelit01.eps, kozelit02.eps stb. alakban kapjuk. A POSTSCRIPT ábrák készítése nagyon hasznos, például ha valamit TEX/LATEX-ben akarunk írni!

[92] A ; esetében kiírja az output-ot: vagy szebb formában, amit beírtunk, vagy ha **plot**-ot (azaz rajzolást) írtunk be, akkor kirajzolja, amit kérünk, és feldolgozza azt.

19.6.2. Szimbolikus számolások, algebrai műveletek

A MAPLE (MATHEMATICA) két dolgot tud nagyon jól: (a) szimbolikusan számolni, azaz formulákat manipulálni és (b) matematikai objektumokat ábrázolni.

Tanulmányaink során gyakran kell bonyolultabb formulákat egyszerűbb alakra hoznunk. Ezt bármely szimbolikus programcsomag könnyedén megteszi. A szimbolikus számolás azt jelenti, hogy míg a gépek korábban csak számokat, vektorokat, stb. tudtak összeszorozni, az általunk leírt programcsomagok algebrai kifejezésekkel, absztrakt szimbólumokkal is számolnak, legalább olyan jól, mint egy matematikus.

Szeretnénk pl. egyszerűsíteni az $(a^6 - b^6)/(a^4 - b^4)$ kifejezést. Beütjük:

> `f:= a^6-b^6;`

Erre azt kapjuk: $f := a^6 - b^6$. Kíváncsiságból faktorizáljuk:

> `factor(f);` Az eredmény:

$$(a-b)\,(a+b)\,(a^2+a\,b+b^2)\,(a^2-a\,b+b^2).$$

Beadjuk:

> `g:=a^4-b^4;`

$$g := a^4 - b^4.$$

A hányadosra alkalmazzuk a **simplify (f/g);** parancsot, mire ezt kapjuk: [93]

$$\frac{a^4+a^2\,b^2+b^4}{a^2+b^2}.$$

Kipróbáljuk a MAPLE „trigonometrikus" erejét is. Beadjuk: **expand(sin(3∗x));**, mire azonnal kifejti $\sin x$ és $\cos x$ hatványai szerint:

$$4\sin(x)\cos(x)^2 - \sin(x).$$

Ezt még mi is ki tudnánk számolni, de a gép az **expand(sin(13∗x));** hatására a $\sin(13x)$-et is egy pillanat alatt kifejti. Egyszerre csak kitágulnak a lehetőségeink.[94]

9.30. ?

9.31. ?

9.30. és 9.31.

A MAPLE egyenleteket is megold. Ha pl. azt írjuk be, hogy

> `solve(x^2-x-1);`

azaz a program oldja meg az $x^2 - x - 1 = 0$ egyenletet, (solve = megoldani) akkor az

$$\frac{1}{2}+\frac{\sqrt{5}}{2},\ \frac{1}{2}-\frac{\sqrt{5}}{2}$$

sort kapjuk vissza: nem tizedestört-közelítést.

Behelyettesítés: Erre szolgál a **subs(mit,mibe)** utasítás.[95] 59 Írjuk be:

59

Sharp curve

> `f=xysin(x+y);`

> `subs(y=3x-1,f);`

Mit kapunk?

[93] Itt most elkezdünk tömöríteni: nem írunk mindent új sorba, nem mindig írjuk ki a gép válaszát. Kiemelt képletnél a sorvégi pontot gyakran elhagyjuk.

[94] Ehhez kapcsolódnak a 9.30. és 9.31. feladatok.

[95] substitute=helyettesít.

19.6.3. Függvényábrázolás a MAPLE segítségével I

Az itt következő, függvényábrázolásról szóló részt – amely bepillantás a MAPLE világába – ismét kedvcsinálónak szánjuk.

Intuíció és képi megjelenítés. A függvényekkel való ismerkedésünkhöz, pl., ha meg szeretnénk sejteni valamit egy függvényről, a MAPLE kitűnően használható: a MAPLE egyebek közt egy nagyerejű függvénykirajzoló program is. A 11.1 alfejezet legtöbb feladatához segítséget nyújthat (bár gyakran gép nékül gyorsabban célhoz érünk). [60]

(60) **Miki: Itt mit írjunk, milyen stílusban?**

A MAPLE számunkra legfontosabb utasítása a **plot**. Durván két szintet különböztethetünk meg, aszerint, hogy be kell-e hívnunk a rajzoló könyvtárat, vagy sem.[96] Mielőtt a rajzolást ismertetnénk, próbálkozzunk egy kicsit.

(61) **Miki: Van ellenpelda erre?**

(b) A MAPLE csaknem [61] minden olyan függvényt ismer, amellyel a könyvben találkozhatunk. [97] Ezekből definiálhatunk további függvényeket.

Definiáláshoz az [:=]-t használjuk, pl. beírjuk:

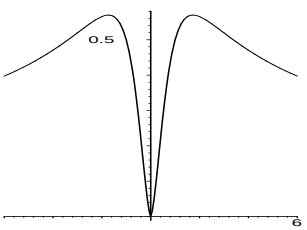
```
> g:=abs(x)^(3/2)/(1+x^2);
```

Az [ENTER] leütése után a program szépen is kiírja, amit beírtunk. Most pl. kiírja, hogy

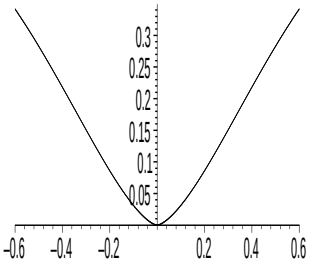
$$g(x) := \frac{|x|^{3/2}}{1+x^2}. \tag{242}$$

Ha ábrázolni szeretnénk, beírjuk:

```
> plot(g,x=-6..6);
```



19.5(a) ábra: $g(x) = \frac{|x|^{3/2}}{1+x^2}$



19.5(b) ábra: Kinagyítva, nem skálázva

A 19.5(a) ábrát kapjuk, kicsit más feliratozással. Figyeljük meg, hogy itt a MAPLE választotta meg a skálázást.

14. FELADAT. Ábrázoljunk néhány, a könyvben található függvényt MAPLE-lel: több ilyet találunk a **??** Példában. [**??** Példában. ?]

15. FELADAT. Ábrázoljuk az $y = x(\pi - x)\sin(8x)$ függvényt MAPLE-lel a $[-2\pi, 2\pi]$ intervallumon, majd a $[0, \pi]$-n. (MAPLE-ben a π beírható Pi-ként!)

[96] Itt még csak a **plot** utasítást használjuk, de a könyvár behívása is nagyon egyszerű: beütjük, hogy **with(plots);**.

[97] Néha nem pontosan ugyanúgy jelöli, pl. $\sin x$ helyett $\sin(x)$-et kell írnunk, $\log x$ helyett $\ln(x)$-et. stb.

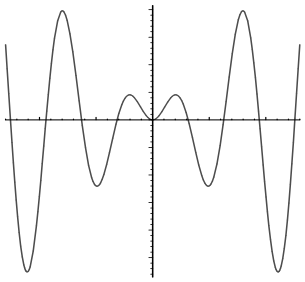
16. FELADAT. Ábrázoljuk az (242) függvényt MAPLE-lel a $[-0.6, 0.6]$ intervallumon (19.4(b) ábra). Ugyanolyan csúcsosnak látjuk-e, mint a 19.4(a) ábrán?

(19.4(b) ábra). ?

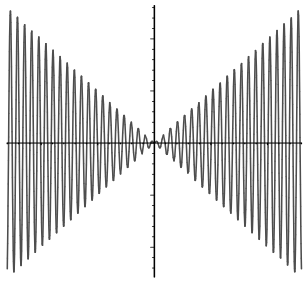
19.4(a) ?

Az alábbi 3 kép pl. a MAPLE program segítségével készült, és az $x \sin x$ függvényt ábrázolja, a $(-13, 13)$, $(-130, 130)$, illetve a $(-65, 65)$ intervallumokon.

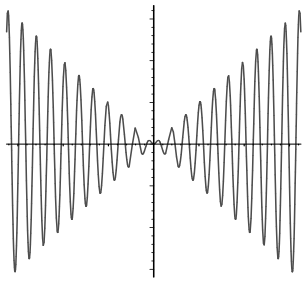
A 19.6(a) ábrán túl rövid intervallumot választottunk, nem láttuk, amit látni akartunk volna. Ezért megtízszereztük az intervallumot, amitől túl sok hullámot kaptunk, és hogy egy kicsit jobban lássuk a részleteket, visszacsökkentettük az intervallum hosszát.



$x \sin \frac{1}{x} \quad x \in [-13, 13]$
19.6(a) ábra

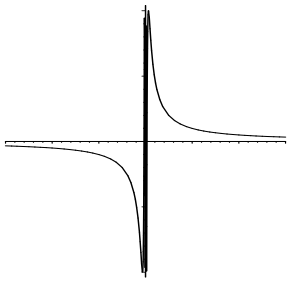


$x \sin \frac{1}{x}$ $x \in [-130, 130]$
19.6(b) ábra

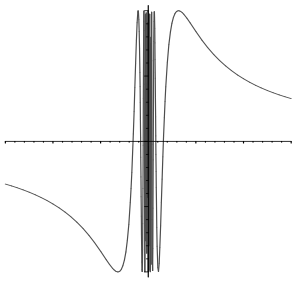


$x \sin \frac{1}{x} \quad x \in [-65, 65]$
19.6(c) ábra

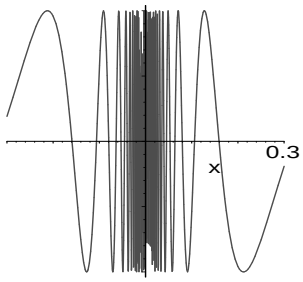
A következő ábrákon a $\sin \frac{1}{x}$ függvényt ábrázoltuk.



$\sin \frac{1}{x}$, $x \in [-30, 30]$
19.7(a) ábra



$\sin \frac{1}{x}$, $x \in [-3, 3]$
19.7(b) ábra



$\sin \frac{1}{x}$, $x \in [-0.3, 0.3]$
19.7(c) ábra

17. FELADAT. Mivel a hullámok a 0 körül nagyon besűrűsödtek, próbáljuk meg átskálázással láthatóbbá tenni 0 környékét: írjunk x helyébe $x/100$-at. Mit tapasztalunk?

Itt tehát ugyanazt a függvényt vizsgáltuk meg „mikroszkópon" keresztül, csak a (vízszintes) nagyítás változott. Hogyan készítettük a fenti ábrákat? Ezt is bemutatjuk majd a következőkben.

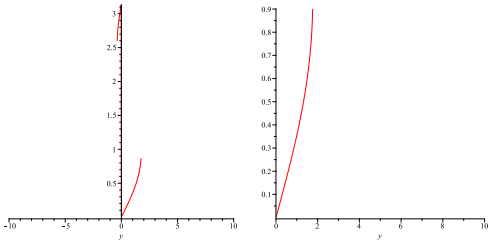
18. FELADAT. A 19.7 ábrán az $x \sin(1/x)$ függvényt vizgálgattuk. Tegyük meg ugyanezt $x^2 \sin(1/x)$-szel is. [62]

(62)

Miki: Bekapcsolni19-hoz?

19. FELADAT. Ábrázoljuk MAPLE segítségével a $\sin(\sin(x))$ függvényt az $[-1,1]$, $[-3,3]$, $[-7,7]$, $[-20,20]$ intervallumokon.[98]

Kimaradt az idevezeto szoveg:



Ezt kapjuk:

A memória törlése: Helyes a MAPLE munkalapjainkat

```
> restart;
```

-tal kezdenünk. Ez törli a memóriát. Erre azért van szükség, mert gyakran egy ilyen lap elején bevezetünk bizonyos paraméterektől függő objektumokat, majd később ezen paramétereknek érteket adunk. Megnézzük, mi történik ezen paraméter mellett, majd a lap elejére ugorva újra végrehajtjuk a lapon található parancsokat. Ha pl. a lap elején szerepel

```
> f:=sin(ax);
```

később pedig az $a := 1$ értékadás, akkor újraindításkor – restart nélkül – ez megmarad: $f = \sin x$ lesz.

19.6.4. Határérték, konvergenciasebesség és a MAPLE

A MAPLE tud határértéket számolni. Írjuk be a MAPLE-be:

```
> limit(n^(1/n),n=infinity);
> limit(n*(n^(1/n)-1),n=infinity);
> limit((n/ln(n))*(n^(1/n)-1),n=infinity);
```

A gép rendre kiírta a 3 határértéket: 1, ∞ és 1. Az utóbbi szerint tehát $\sqrt[n]{n} \sim 1 + \frac{\log n}{n}$. A MAPLE tehát remekül számol határértéket is.

Beírjuk:

```
> limit((1-1/n)^n,n=infinity);
```

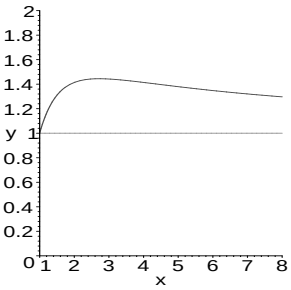
és a gép kiírja az $1/e$-t.

Jó ez nekünk? Igen is, meg nem is. Jó, ha magunk nem találtuk volna ezt ki, de miután a gép megmondta, már be tudjuk bizonyítani. Nem jó, ha a válasz után semmivel sem tudunk többet a kérdésről, nem lettünk tőle okosabbak. Lesznek, akiknek a végeredmény önmagában is hasznos, de a témával most ismerkedők számára fontos tudni, hogy vannak olyan parancsok is, amelyek hatására a MAPLE elárulja, hogyan is csinálta.

Most a MAPLE segítségével újra „megtámadjuk” iskolafeladatunkat, de egy kicsit másként. Újra arra vagyunk kíváncsiak, mihez tart $\sqrt[n]{n}$, és milyen gyorsan. Fogjuk

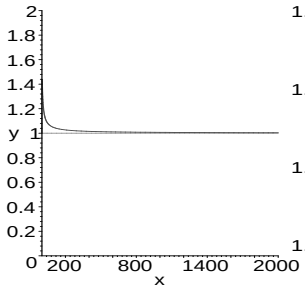
[98] Az a meglepő, hogy amit kapunk, hasonlít a $\sin x$ görbéjéhez.

fel a sorozatot úgy, mint az $f(x) = x^{1/x}$ függvény értékeit az $x = n$ sorozat mentén. Ha MAPLE-ben ábrázoljuk az $x^{1/x}$-et, a következő adódik.



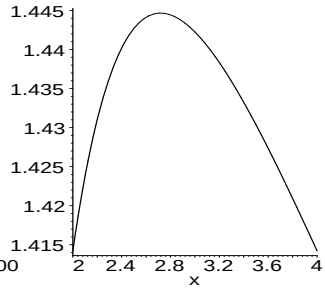
$x^{1/x}$ $x \in [1, 8]$

19.8(a) ábra



$x^{1/x}$ $x \in [1, 2000]$

19.8(b) ábra

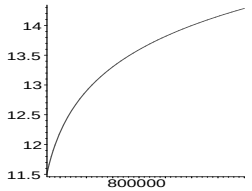


$x^{1/x}$ $x \in [2.4]$

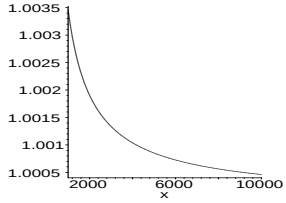
19.8(c) ábra

Ebből valóban jól látható, hogy a függvény eleinte növekszik, majd csökken, és (nem túl gyorsan) 1-hez tart. A 19.8(c) ábra azt mutatja, hogy a függvény valahol az $x = e$ pont közelében éri el a maximumát.

(Amikor a függvénydiszkussziót tanuljuk, akkor ennek igazolása könnyű feladat: csak azt kell tudni, hogy $f(x)^{g(x)}$ helyett gyakran érdemes a logaritmusát vizsgálni.)



19.9(a) ábra: $(x^{1/x} - 1) \cdot x$



19.9(b) ábra: $(x^{1/x} - 1) \cdot \frac{x}{\log x}$

Mármost a hiba nagyságrendjének megállapítása mehet „felezéses-próbálgatásos eljárással", de nagyon óvatosnak kell lennünk. A két kapott függvény tükrözi ezt. A 19.9(a) függvénye végtelenhez tart (legalábbis úgy látszik), amíg a 19.9(b) ábrán

$$(x^{1/x} - 1) \cdot \frac{x}{\log x} \to 1.$$

Az olvasó joggal lehet kíváncsi, hogy a konvergenciasebesség, amit a gépen próbáltunk megsejteni, matematikailag hogyan kezelhető. A válasz az, hogy ott, ahol ez felvetődött, ott még nem kezelhető, viszont amikor már tudjuk, hogy $(e^x)' = 1$ az $x = 0$-ban, akkor könnyűvé válik; lényegében ezzel ekvivalens. Vázolunk egy megközelítést: a fentiből $\frac{e^u - 1}{u} \to 1$, ha $u \to 0$. Így $n = x = 1/u$-t helyettesítve

$$\sqrt[x]{x} - 1 = x^{\frac{1}{x}} - 1 = \frac{1}{u^u} - 1 = e^{-u \log u} - 1 \sim -u \log u = \frac{\log x}{x}.$$

(Lásd a 10.2/33.(e) feladatot.) [63]

(63)
Miki: Kati: Hogyan kezeljem ezeket a kettos referenciakat. Gondolom, Te 1 referenciaval is elkezeled oket.

19.6.5. Rajzolás, függvényábrázolás MAPLE-lel II

A MAPLE sok programkönyvtárral rendelkezik. Bonyolultabb utasításokhoz ezeket be kell hívni. [99] Itt három könyvtárat említünk (a with-et elhagyhatnánk):

with(plots)	rajzoláshoz	pl. felületek, ...
with(linalg)	lineáris algebrához	pl. mátrixok
with(curvefitting)	polinom-approximáció, interpoláció	Bernstein, Lagrange

with(plots):
Görbék, felületek kirajzolásával kapcsolatban itt három parancsot használunk:

plot, implicitplot, plot3d.

Az utóbbi kettőhöz be kell hívnunk a rajzoló ,,könyvtárat", be kell ütnünk, hogy **with(plots).** [100] Mint már említettük, ha a ; helyébe : -ot írunk, akkor a MAPLE elhagyja a ,,visszajelzést".

Magához a **plot**-hoz nem kell a könyvtár. De ha beütjük:

```
> with(plots);
```

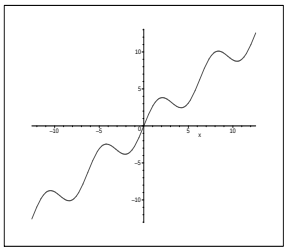
akkor a MAPLE kiírja lehetséges grafikus parancsokat.

Az értelmezési tartomány és értékkészlet megszorítása

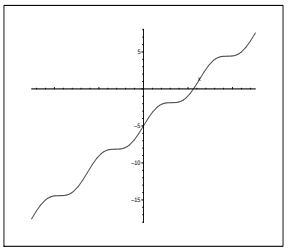
Függvényábrázoláskor be kell írnunk, mettől meddig akarjuk a függvényt ábrázolni. A

```
> plot(x+2*sin(x),x=-12..12);
```

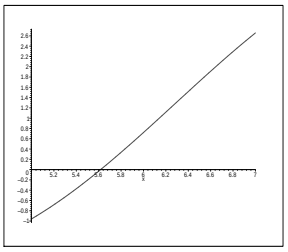
kirajzolja az $x+2\sin x$ függvényt -12-től 12-ig. Az eredmény a 19.10(a) ábrán látható:



$x \in [-7, 14]$
19.10(a) ábra: $x + 2\sin x$,



$x \in [-7, 14]$
19.10(b) ábra: $x + \sin x$,



$y = -0.1..0.1$
19.10(c) ábra: a gyök ≈ 5.6

Néha csak egy adott téglalapon belül szeretnénk a függvényt ábrázolni. Most az $y = x + c \cdot \sin x$ ábrázolásába kezdünk, $c = 1/2,\ 1,\ 2$-re. Mondjuk, $x + \sin x = 5$ gyökeire vagyunk kíváncsiak. Beírjuk:

```
> plot(x+sin(x)-5,x=-12..12);
```
[64]

Az eredmény a 19.10(b) ábrán látható. Beadjuk[101]

```
> plot(x+sin(x)-5,x=5..7,y=-1..2.6);
```

Az eredmény a 19.10(c) ábrán látható.


(64)
Miki: Itt a sor -12..12-je nincs osszhangban az abran lathato ertelmezesi tartomannyal.

20. FELADAT. Ábrázoljuk $x + c \cdot \sin x$-et c különböző értékeire. Mely c-kre látszik az eredmény monoton növőnek? (Miért?)

[99] Annak elsősorban technikai okai vannak, hogy induláskor a MAPLE miért nem hívja be eseket mindet: akkor nem kellene foglalkoznunk velük.

[100] Innentől kezdve a **,,kiírja"** részt gyakran elhagyjuk!

[101] Az ábrákat a kinyomtathatóság végett kicsit átalakítjuk: megvastagítjuk a vonalakat, felnöveljük a beírásokat, és ahol zavaróvá válnak, ott elhagyjuk őket.

Megjegyzés 19.2 *Persze, a* $x + \sin x = 5$*-öt a*

```
> evalf(x+sin(x)-5);
```

is megoldja: azonnal kiírja, 5.617555005.

A plot további kapcsolói:

Ha két vagy három (vagy akárhány) görbét akarunk egyazon koordináta-rendszerben kirajzolni, akkor szögletes zárójelek közé tesszük őket, pl.

```
> plot([f,g],x=a..b);
```

alakban. Ha beütjük, hogy

```
> plot([cos(x),sin(x)],x=-4..4);
```

a 19.11. ábrán látható ábrát kapjuk, az egyik görbét pirosan, a másikat zölden. Nagyon fontos, hogy a számítógépes eredményeknél befolyásolhassuk a végeredmény megjelenítését.

```
> plot([cos(x),sin(x)],x=-4..4,color=black);
```

már fekete ábrát fog adni, a

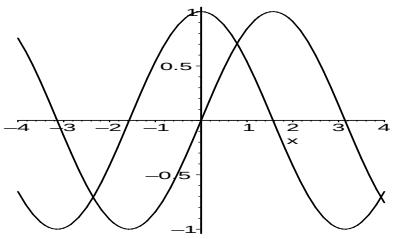
```
> plot([cos(x),sin(x)],x=-4..4,color=black,thickness=3);
```

a görbél vastagságát fogja megháromszorozni.

21. FELADAT. Próbáljuk ki az alábbi két sort:

```
> plot([sin(x),x-x^3/6],x=-4..4,color=black)

> plot([sin(x),x-x^3/6],x=-4..4,y=-0.5..1,color=black)
```



19.11. ábra: két fv. egyszerre

22. FELADAT. 5.2/29 feladat általánosabban úgy fogalmazható, hogy ha

$$f_a(x) = \left(1 + \frac{1}{x}\right)^{x+a}$$

akkor ez $a = 0$-ra és $a = 1$-re ,,jó ismerősünk", (legalábbis, ha $x = n$ egész.) Az első esetben alulról, a második esetben felülről tart e-hez. De mi van közben? Bizonyítsuk be hogy

(a) $a \leq 1/2$-re monoton növekedve tart e-hez,

(b) $a > 1/2$-re monoton csökkenve tart e-hez.

[65]

Ha ezt a MAPLE-vel akarjuk ,,megvizsgálni", a következő sorokra van szükségünk:

```
> restart;
```

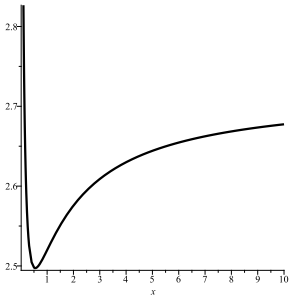
(65)

Miki: Azt hiszem, ezt a Polya-Szegobol vettem. Nem volna-e erdemes betenni az irodalomba?

```
> f:=(1+(1/x))^(x+a));
> plot(subs(a=1/3,f), x=0.1..10);
```
[66]

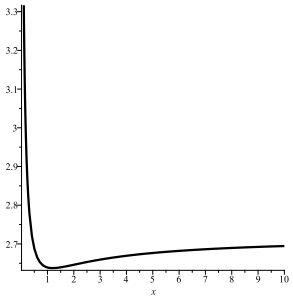
(66)
Miki: Javitva

A következő ábrákat kapjuk, ha $a = 1/3$. $a = 2/5$ és $a = 1/2$-t helyettesítünk be a fenti képletbe (a jobb láthatóság végett a vonalvastagságot és a színt is megváltoztattuk).



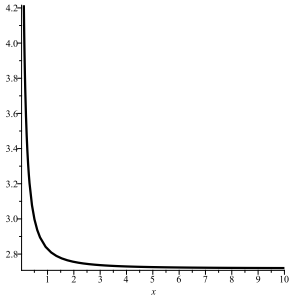
a=1/3

19.12(a) ábra



a=0.4

19.12(b) ábra



a=1/2

19.12(c) ábra

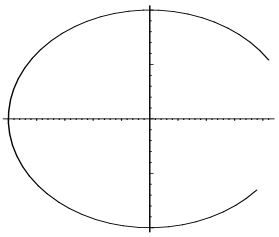
←— **Itt tartok** C

19.6.6. Paraméteres görbe kirajzolása

Adott $f(t), g(t)$ függvényekre próbáljuk ki, mi történik, ha a] zárójelet „rossz" helyre tesszük:

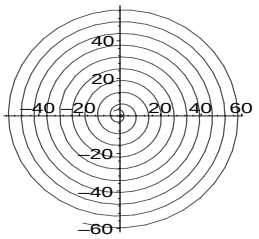
```
> plot([f,g,t=a..b]);
```

Ilyenkor az $x = f(t), y = g(t)$ „pályát" rajzolja ki a MAPLE, $t \in [a, b]$-re. Ezzel is találkozunk még: ez matematikában is és fizikában is nagyon fontos. Így kapjuk az alábbi ábrákat, a MAPLE által skálázva.



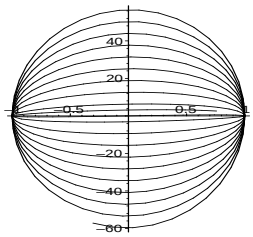
Körív:
$(\cos(t), \sin(t))$,
$t \in [-4, 1]$

19.13(a) ábra



Spirál:
$(t\cos(t), t\sin(t))$,
$t \in [-4, 60]$

19.13(b) ábra



Lemaradt egy t!
$(\cos(t), t\sin(t))$,
$t \in [-4, 60]$

19.13(c) ábra

19.6.7. Differenciálás

A MAPLE segítségével differenciálni is tudjuk a függvényeket (bár az eredményt néha nem abban a formában kapjuk meg, amelyben szeretnénk) [102]. Lássuk pl. az

[102] Ha akkor érsz ezen sorokhoz, amikor még nem tanultál differenciálni, ugord át.

$f := b^{(x^2)} \cdot \sin(x)$ függvényt:

```
> diff(f,x);
```

$$2\,b^{(x^2)}\,x\,\ln(b)\,\sin(x) + b^{(x^2)}\,\cos(x).$$

23. FELADAT.

"Ellenőrizzük" a 10.2 fejezet tételeit (pl. 10.27-10.31 tételeket) MAPLE segítségével. [67]

(67)
Miki: feladat, feladatok?

19.6.8. Racionális törtfüggvények

Vizsgáljuk meg az

$$y = \frac{x^3 + x^2 - 7x}{x^2 - 3x + 1}$$

racionális törtfüggvényt. Érdemes a nevező gyökeinek megkeresésével kezdeni. Beütjük $f := x^3 + x^2 - 7x$-et és $g := x^2 - 3x + 1$-et, majd a **solve(g=0);**-val folytatjuk. Az eredmény:

$$\frac{3}{2} + \frac{\sqrt{5}}{2}, \qquad \frac{3}{2} - \frac{\sqrt{5}}{2}.$$

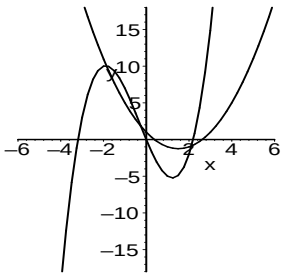
Minket a tizedestört-kifejtés is érdekel. Beütjük: **evalf(solve(g=0));** ahol az **evalf** az evaluate-float-ot rövidíti (evaluate = kiértékelni). Az eredmény:

2.618033988, 0.381966012

Most ábrázoljuk előbb f-et és g-t külön, majd f/g-t a

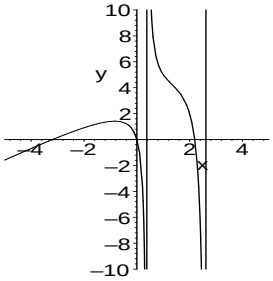
```
> plot(f/g,x=a..b,y=c..d);
```

utasítással, az ábrán látható a, b, c, d-kel.



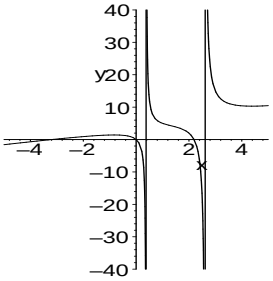
plot([f,g],x=-6..6,y=-18..18);

19.14(a) ábra



plot(f/g,x=-5..5,y=-10..10);

19.14(b) ábra



plot(f/g,x=-5..5,y=-40..40);

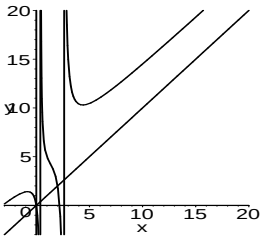
19.14(c) ábra

A 19.14(b) ábrán azért kellett az $y \in [-10, 10]$-zel levágnunk az ábrát, hogy bizonyos részleteket ne nyomjon agyon az y irányú skálázás. (Próbáljuk elhagyni az $y = -10..10$-et vagy helyettesíteni $y = -2000..2000$-rel.) Így a nevező 2. gyöke fölötti részletek tűnnek el. Ha $y \in [-40, 40]$-nel vágunk, a 19.14(c) ábrát kapjuk.

Az itt megadott ábráink szépek, részben mert a vonalvastagságot valójában megnöveltük. Az ábrabeírásokat és a tengelyek beosztását is meg tudjuk változtatni.

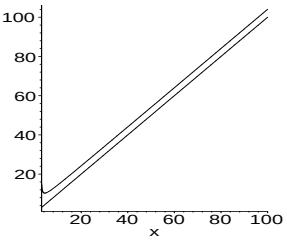
Kifogásolható az is, hogy nem azonos a skálázásunk az x és az y irányban. Ezen könnyen segíthetünk, ha mindkét irányban ugyanolyan hosszú intervallumot adunk meg.[103] Ilyenkor azonban fontos részletek tűnhetnek el.

Szeretnénk megtudni, hogyan viselkedik függvényünk a második gyök ($x = 2.618$) körül. Ezért kirajzoljuk $[-3, 20]$-ban is. Most y-t is ugyanígy vágjuk.



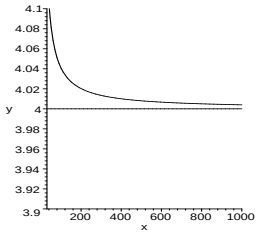
plot(f/g,x=-3..20, y=-3..20);

19.15(a) ábra



plot([f/g,x],x=3..100);

19.15(b) ábra



plot((f/g)-x,x=30..1000);

19.15(c) ábra

Mi van nagy értékekre? Elkészítettük a 19.15/b ábrát, amiből arra következtethetünk, hogy $y = x$ aszimptotája a görbének. Hogy ezt jobban megvizsgáljuk, a 19.15/c ábrán a különbségüket vettük. Ebből arra következtettünk, hogy $f/g - x \to 4$ ha $x \to \infty$. Hogy ezt jobban lássuk, (utólag) megszorítottuk y-t $[4, 4.2]$-re.

Könnyű megmutatni, hogy ez a sejtés valóban helyes, azaz

$$\lim_{x\to\infty} \left(\frac{x^3 + x^2 - 7x}{x^2 - 3x + 1} - x \right) = 4.$$

(Lásd a 9.2 Tételt.)

24. FELADAT. "Oldjuk meg" az 111. oldalon szereplő, $\sqrt{x^2 + ...}$ aszimptotájára vonatkozó feladatot a fenti stílusban.

19.6.9. A MAPLE, függvényábrázolás, és az egyenlőtlenségek

Képesek vagyunk egy ábrára több függvényt is kirajzolni egyszerre. Ki tudjuk nagyítani ábráinkat a kritikus helyek környékén. Így egyenlőtlenségeket is vizsgálgathatunk. Láttuk, ha pl. az f, g és h függvényeket szeretnénk kirajzolni egyszerre, akkor a **plot([f,g,h],x=a..b);** alakot használjuk.

A matematikai bizonyításokban az egyenlőtlenségek fontos szerepet játszanak. Ezt láttuk már pl. a Bernoulli-egyenlőtlenség esetében, a Jensen-egyenlőtlenségnél és még sok más esetben. Segíthet-e a MAPLE (vagy a MATHEMATICA) egyenlőtlenségek kezelésében? Igen, pl. kirajzolhatunk függvényeket egyazon koordinátarendszerben, és rögtön látjuk, hogy egyik nagyobb-e, mint a másik. Az alábbiakban ezt mutatjuk be, kicsit leegyszerűsítve.

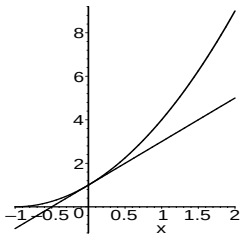
A Bernoulli-egyenlőtlenség. (l. 2.5 Tétel.)
Beírjuk a MAPLE-be:

[103] Segít ezen a **scaling = constrained** is.

MAPLE MUNKALAP 1

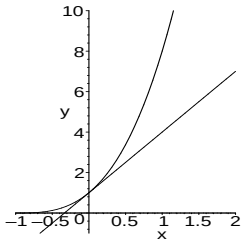
```
> f:=(1+x)^n;                          (f = (1+x)^n)
> g:=1+n*x;                          (a lineáris közelítés)
> fn:=subs(n=3,f);                (subs=substitute = behelyettesít)
> gn:=subs(n=3,g);
> plot([fn,gn],x=-1..2,y=-1..10,color=black);          (kirajzol)
```

Fent tehát megadtuk a Bernoulli-egyenlőtlenség mindkét oldalát, (f, g)-t, majd behelyettesítettünk $n = 3$-at, kaptuk a 19.16(b) ábrát. Ha $n = 2$-t helyettesítünk, a 19.16(a) ábrát kapjuk, $n = 5$-re a 19.16(c) ábrát. A fenti utolsó MAPLE-sorban



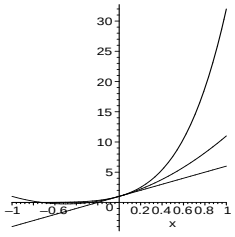
$(1+x)^2$, $1+2x$

19.16(a) ábra



$(1+x)^3$, $1+3x$

19.16(b) ábra



$(1+x)^5$, $1+5x$, $1+5x+10x^2$

19.16(c) ábra

az $y = -1..10$ elhagyható, de nélküle a MAPLE saját maga dönti el a függőleges nyújtást. Mit csináltunk eredetileg? Csak a -1 és 10 közötti értékeket kértük.

[68] Hopp!!! A valóság az, hogy az utolsó ábra nem volt túl informatív, így kisebb intervallumot vettünk, és behoztuk a másodfokú közelítést is: lecseréltük az utolsó parancsot az alábbi parancsra:

(68)

Miki: A Hopp es kornyeke ellenorzendo, Mick nem erti

```
> plot([f,g,h],x=-1..1,color=black,thickness=3);
```

ahol $n = 5$-re $f(x) = (1+x)^n$, $g(x) = 1+nx$ és $h(x) = 1+nx+\binom{n}{2}x^2$. Mit láthatunk a kapott 19.16(c) ábrán?

19.6.10. Polinomközelítés (lokális)

Az itt következő téma, ahogyan ezt a 11.7 tétel tájékán is láttuk, még folytatása az egyenlőtlenségek tárgyalásának, ugyanakkor átvisz a polinomközelítésre: függvényeket fogunk közre hozzájuk közeli polinomokkal.

Fontos számunkra, hogy

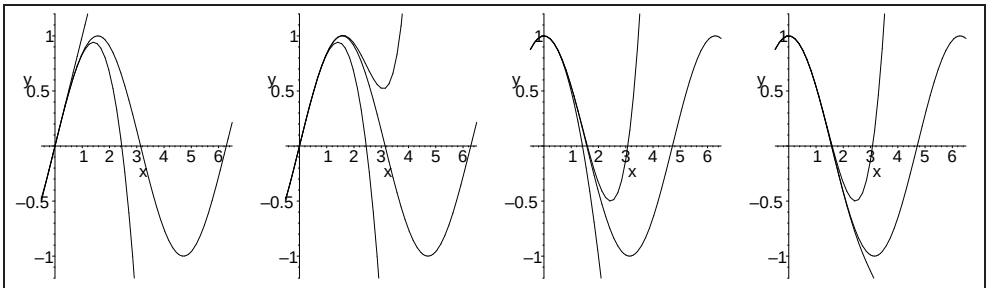
$$x - \frac{1}{6}x^3 < \sin x < x, \qquad \text{ha} \qquad x > 0.$$

25. FELADAT. Mutassuk be ezt a MAPLE segítségével.

A fenti feladat továbbvitele:

26. FELADAT. Mutassuk be a MAPLE segítségével a következőt:

$$x - \frac{1}{6}x^3 < \sin x < x - \frac{x^3}{6} + \frac{x^5}{120}, \qquad \text{ha} \qquad x > 0.$$



17. ábra: (a) (b) (c) (d)

Fuggvenyek es Taylor polinomjaik: $\sin x$, illetve $\cos x$ közrefogása Taylor polinomjaikkal

$\sin x$: $x - \frac{x^3}{6}$, $x - \frac{x^3}{6} + \frac{x^5}{120}$,
$\cos x$: $1 - \frac{x^2}{2}$, $1 - \frac{x^2}{2}$, $1 - \frac{x^2}{2} + \frac{x^4}{24}$

A 19.17(a)–(b) ábrán először $\sin x$-et x és $x - \frac{x^3}{6}$ fogja közre, majd $x - \frac{x^3}{6}$ és $x - \frac{x^3}{6} + \frac{x^5}{120}$. Hasonlóan a 19.17(c)–(d) ábrán $\cos x$-et fogja közre előbb $1 - \frac{x^2}{2}$ és $1 - \frac{x^2}{2} + \frac{x^4}{24}$, majd $1 - \frac{x^2}{2} + \frac{x^4}{24}$ és $1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720}$.

27. FELADAT. Mutassuk be MAPLE segítségével,[104] hogy

$$1 - \frac{1}{2}x^2 < \cos x < 1 - \frac{1}{2}x^2 + \frac{1}{24}x^4, \qquad \text{ha} \qquad x > 0.$$

Az ábrázolás néha segíthet bizonyításainkban, de azokat nem pótolja. Az idevágó bizonyítást illetően lásd a 202. oldalon a 92. feladatot.

[69] A 11.8 Tételben láttuk, hogy alkalmas feltételekkel

$$\lim_{n\to\infty} \sum_{k=0}^{n} \frac{f^{(k)}(0)}{k!} x^k = f(x), \qquad \text{ha} \qquad n \to \infty.$$

(69)
Miki: Ellenorzendo a Taylor hivatkozas

A 11.2. alfejezetben szerepeltek a Taylor-polinomokkal és Taylor-sorok. (L. 11.6 Definíció, 11.7 Tétel.) A MAPLE-ben megkaphatjuk a függvények Taylor-polinomjait. Ha pl. a $\log(1 + x)$ harmadik, $x = 0$ körüli Taylor-polinomját akarjuk megkapni, beírjuk:

```
> f := ln(1+x);
> g:=taylor(f,x=0,4);
```

hatására a MAPLE kiadja:

$$g := x - \frac{1}{2}x^2 + \frac{1}{3}x^3 + \mathrm{O}(x^4).$$

Itt az $O(x^4)$ tagban az ordo jelet a 114. oldalon definiált értelemben használjuk. Az állítás a 11.7 Tételből következik. [70]

(70)
Miki: Megkerdezni V-M-t, mire hivatkozzak inkabb?

[104] Ez BASIC pogrammal is egyszerűen megtehető.

Az így kapott formulával kicsit óvatosabban kell bánnunk. Ha pl. az eredményül kapott g függvényt ki akarjuk rajzolni, akkor az $O(\,.\,)$ tagot el kell hagyni.

28. FELADAT. A 211. oldalon és környékén több Taylor sorfejtést találunk. Állítsuk elő ugyanezeket a Taylor sorfejtéseket a MAPLE segítségével.

Van a MAPLE-nek egy szinte ijesztő, újszerű alkalmazása: számtalan elméleti feladatot, pl. a könyv nagyon sok elméleti feladatát meg tudja oldani, bizonyos értelemben. Ha pl. szeretnénk egy zárt formulát a $\sum_{k=0}^{n} k^2$-re, elegendő beírnunk a MAPLE-be:

$>$ `sum(`k^2`, k=0..n)` és a gép kiírja a keresett formulát. [71]

Miki: Ez uj formatumu, itt dontes kell!

19.7. MAPLE és a "komolyabb" kérdések

19.7.1. Határozatlan integrál, primitív függvény

A primitív függvényt is kiszámolja a MAPLE (ha lehet). Ezt a $\log x$ és az $x\sin(x)$ [72] primitív függvényének példáján mutatjuk be. (A MAPLE átírja $\log x$-et $\ln(x)$-re!) A primitív függvény kereséséről a 11.3. fejezet *A határozatlan integrál* című alfejezetében volt szó. Az f x-től függő függvény határozatlan integrálját **int(f,x)** adja meg:

Miki: log x v. log(x)?

[73]

Miki: Tablazat meretezese

	Ezt írtuk be	Ezt adta vissza	Jelentése
1a.	$>$ **g := ln(x);**		egy g fv. definiálása
1b.		$\ln(x)$	
2a.	$>$ **int(g,x);**		Keressen primitív függvényt.
2b.		$x\ln(x) - x$	
3a.	$>$ **int(x∗sin(x),x);**		Keressen primitív függvényt.
3b.		$\sin(x) - x\,\cos(x)$	

19.7.2. Még mit érdemes tudnunk a MAPLE-ről?

Kiegészítés és ismétlés

Kulcsszó/Példa	Megjegyzés	Értelmezés
expand(. . .)		kifejt
plot(f,x=a..b)		ábrázolja $f(x)$-et $x \in [a,b]$-re
z=plot3d(f,x=a..b,y=c..d)	with(plots)	ábrázolja $f(x,y)$-t $x \in [a,b]$, $y \in [c,d]$-re
simplify(. . .)		egyszerűsít
solve(f=0,x)		megold egy $f = 0$ egyenletet x-ben
subs(x=t^2,g)	substitute	függvénybe behelyettesítünk $x = t^2$-et
evalf	evaluate float	tizedestört alakba írunk. . . (kiértékelünk)
scaling=constrained	skálázás	letiltja az átskálázást
font=[HELVETICA,18]		BETŰMÉRET=18
font=[TIMES,ROMAN,24]		típus=ROMAN,. . .

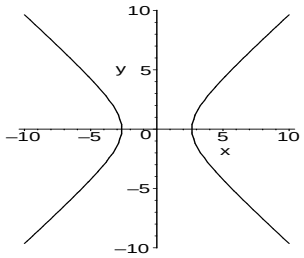
Az alábbiakkal kicsit előreugrunk, a többváltozós függvényekre.

19.7.3. Implicitplot

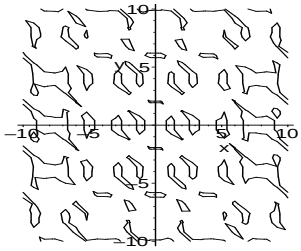
105

Az **implicitplot** azt jelenti, hogy beírunk egy $f(x, y) = c$ alakú egyenletet, és a program kirajzolja a megfelelő pontok mértani helyét. Miért is van rá szükségünk? Mert gyakran bukkanunk olyan egyenletre, amelyet nem tudunk megoldani. Írjuk be:

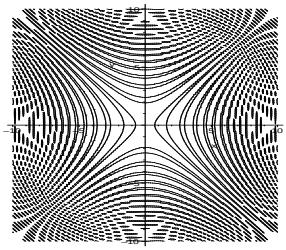
```
> implicitplot(x^2-y^2=7,x=-10..10,y=-10..10,color=black);
```



hiperbola
19.18(a) ábra:



hiperbolasereg
19.18(b) ábra: durva rajz



hiperbolasereg
19.18(c) ábra: finom rajz

Itt tehát a 19.18(a) ábra az $x^2 - y^2 = 7$ kirajzolása. Szerettünk volna görbesereget generálni, így beírtuk a programnak, hogy **implicitplot**-tal ábrázolja a $\sin(x^2 - y^2) = 0.5$ „mértani helyet". Ez adta a 19.18(b) ábrát. Nagyon érdekes, de világos, hogy valami nem jó rajta. Túl kevés pontot használt a rajzoláshoz. Beírtuk:

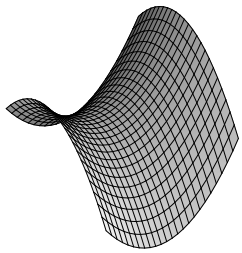
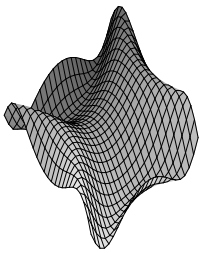
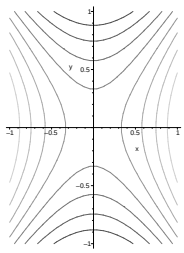
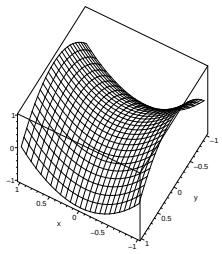
```
> implicitplot(sin(x^2-y^2)=0.5,x=-10..10,y=-10..10,numpoints=2000);
```

azaz használjon sokkal több pontot. Észrevehetően lelassult a gép, de javult az ábra. Végül **numpoints=16000**-rel, rövid várakozás után a 19.18(c) ábrát kaptuk: a keresett (két) hiperbolasereget.

19.7.4. Plot3d

A MAPLE-lel felületeket is rajzolhatunk, erre használjuk a **plot3d** parancsot.

[105] Ez tulajdonképpen a 428–430. oldalon lévő részek befejezése, de mivel többváltozós anyag, idehelyeztük.

19. ábra: (a) (b) (c) (d)

Itt a $z = (x^2 - y^2)$, illetve a $z = \sin(x^2 - y^2)$ 3-dimenziós felületet ábrázoltuk az $x \in [-2, 2]$, $y \in [-2, 2]$ négyzet felett (majd az egérrel elforgattuk).
(**plot3d(x^2-y^2,x=-2..2,y=-2..2);**)

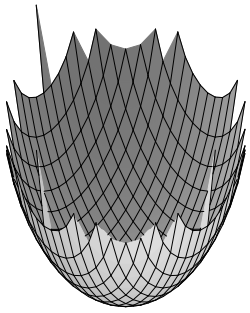
Itt az $(x^2 - y^2)$-et szintvonalasan, illetve kockába zárva rajzoltuk ki.

[74]

(74) **Miki: Ellenorizni a negyzet meretet**

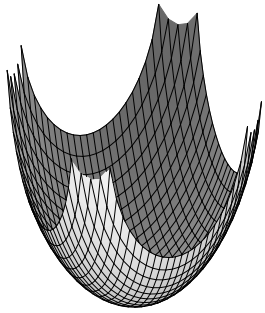
A háromdimenziós egységgömböt az (x, y, z) háromdimenziós koordináta-rendszerben az $x^2+y^2+z^2 = 1$, egyenlet, ennek megfelelően, az alsó félgömböt a $z = -\sqrt{1 - x^2 - y^2}$ függvény írja le. A 19.20. ábrán ezt a félgömböt rajzoltunk volna ki.

Impozáns, de látjuk, a szélei túl szaggatottak. Ezt használtuk:



$-\sqrt{1 - x^2 - y^2}$
Alapértelmezés

19.20(a). ábra



$-\sqrt{1 - x^2 - y^2}$
$x \in [-0.85, 0.85], y \in [-0.85, 0.85]$,
grid=[30,30]

19.20(b). ábra

[75]

(75) **Miki: Ellenorizni a 0.8 vs 0.85-ot**

A bajt az okozta, hogy az $x^2+y^2 = 1$ körvonal felett a felület érintősíkja függőlegessé válik. Ez jól látható, ha csak $x \in ([-0.85, 0.85], [-0.85, 0.85])$ [76] felett ábrázoljuk a felületet, majd rákattintva az egérrel az ábrára, azt elforgatjuk. A finomabb (30×30) rács is segíthet. Az eredmény a 19.20(b) ábrán látható. [77]

(76) **Miki: Aszim intevallum volt itt (?)**

19.8. Mit tud még a MAPLE?

(77) **Miki: Ez a hosszufugg-bol jott ide, meg ellenorzendo**

Mindent tud, amire egy átlagos matematikusnak szüksége lehet.

Tud mátrixokat szorozni, azok inverzét megkeresi, sajátértékeit kiszámolja, karakterisztikus polinomjait megadja, egyenleteket, egyenletrendszereket, differenciálegyenleteket old meg, és még sok olyasmit tud, ami a magasabb matematikához tartozik.[106] Mindezeket nagyon röviden bemutatjuk [2]-ben.

Itt csak analízisre szorítkozunk. De ott is nagyon sokat tud. Tud pl. parciális törtekre bontani, a **parcfrac** segítségével. (Erre az integrálásnál van szükségünk.) Tekintsűk a következő Munkalapot.

MAPLE MUNKALAP 2 (PARCIÁLIS TÖRTEKRE BONTÁS)

```
> restart;
> ractort := (x^2-1)/(x^3+5*x^2+8*x+6);
> convert(ractort,parfrac);
```

Ahol a MAPLE által válaszként megadott szép formákat kihagytuk. Az utolsó sor hatására a gép kiadja

$$\frac{1}{5}\,\frac{-3\,x-7}{x^2+2\,x+2}+\frac{8}{5\,x+15}.$$

19.8.1. Lehet-e MAPLE-ben programokat írni?

Igen, lehet. Minden, ami BASIC-ben megcsinálható, az többnyire MAPLE-ben is megcsinálható.[107] Persze, első kérdés, hogy **mi program és mi nem az**. Itt a programot azzal definiálom, hogy van benne ciklus és/vagy elágazás.

Tehát ,,megmutatjuk", hogyan kell MAPLE-ben ciklust képezni:

29. FELADAT. Próbáljuk ki a következő MAPLE programot:

```
> for i from 1 to 13 do; i*i; end do;
```

19.8.2. Globális polinomközelítés

Mikor használjunk komolyabb programcsomagokat? Ha függvényeket akarunk számítógéppel megközelíteni, gyakran egyszerű programokat írhatunk a probléma megoldására. Alább egy olyan problémát mutatunk be, ahol egyértelműen látható, hogy BASIC-ben és MAPLE-ben egyaránt megoldhatjuk, de a BASIC program megírása sok időt vehet igénybe.

A Bernstein-polinomok. Ezeket (156) definiálta, a 214. oldalon. Emlékeztetünk, hogy az f függvény $[0,1]$-beli n-edfokú Bernstein-polinomja

$$g(x)=B_n(x;f)=\sum_{k=0}^{n}\binom{n}{k}f\left(\frac{k}{n}\right)x^k(1-x)^{n-k}. \tag{243}$$

Az alábbiakban két függvény 8-adfokú közelítését rajzoljuk ki, de az alábbi ,,munkalapon" a függvényt átírva bármelyik másik függvény tetszőleges fokú polinom-közelítését

[106] Ha valamire szükségünk van, ami itt nincs felsorolva, érdemes a HELP-ben ,,rákeresni".

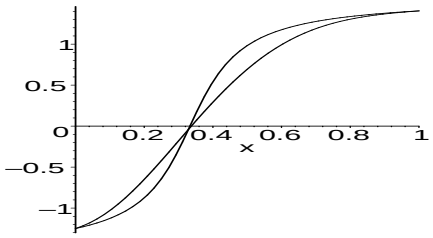
[107] Ezzel szemben a BASIC nem ismeri a szimbolikus műveleteket: nem tud differenciálni, integrálni, gyököt keresni stb. Ezen felül sok mindent, ami a MAPLE-ben adott, a BASIC-ben magunknak kell kitalálnunk, megcsinálnunk.

is kiszámolhatjuk.[108] A 19.21(a) ábrán $\operatorname{arc\,tg}(9x-3)$ és a Bernstein-polinomja, a 19.21(b)-n pedig $g = |x - 0.4|$ és a Bernstein-polinomja látható. (Készakarva nem (1/2)-re szimmetrikus függvényt választottunk, mert az nem mutatná, ha x-et és $(1-x)$-et tévedésből felcseréltük volna.)

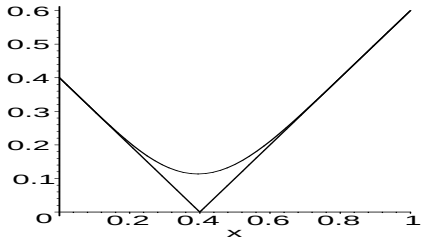
Az alábbi MAPLE Worksheet-et (bernstein.mw) írtuk erre a célra:

MAPLE MUNKALAP 3

```
> restart:                                              (memóriatörlés)
> f:=abs(t-.4):                                         (függvény megadása)
> g:=sum(binomial(n,k)*x^k*(1-x)^(n-k)*subs(t=k/n,f),k=0..n):
> h:=subs(n=8,g):                                       (n = 8-adfokúval approx.)
> plot([h,abs(x-.4)],x=0..1,color=black);               (kirajzoljuk g, h-t.)
```



19.21(a) ábra: $\arctan(9x-3)$, $x \in [0,1]$



19.21(b) ábra: $|x - 0.4|$

Magyarázat

1. Megismételjük: ha egy ,,munkalapon" valamit átírunk és az átírt változattal szeretnénk tovább számolni, akkor újraindítunk: célszerű a **restart**-ra ráállni a kurzorral és nyomogatni az ENTER-t: ez azért kell, hogy először a memória törlődjék: tiszta memóriával induljunk.Ez a MAPLE használatánál nagyon fontos: ha erről elfelejtkezünk, helytelen eredményt kaphatunk. (Van egy Restart gomb is!)
2. A **subs(x=..,f(x,y))** behelyettesítés, az összeget a **sum** és az $\binom{n}{k}$-t a **binomial(n,k)** parancsokkal állíthatjuk elő.
3. A 2. sorban adjuk meg a függvényt.
4. A 3. sor állítja elő a Bernstein-polinomot (243) alapján. Ebben a **sum(u(k),k=0..n);** a $\sum_{k=0}^{n} u(k)$-val ekvivalens. A **subs(t=k/n,f)** pedig azt jelenti, hogy f-be helyettesítsünk $t = \frac{k}{n}$-et.
5. A következő sorban döntjük el, hányadfokú polinommal approximálunk.
6. Az utolsó sor kirajzolja az eredeti függvényt és a közelítését.

A fenti ,,program" más függvényekre is működik: a 2. és az utolsó sorában kell **abs(x-.4)**-et lecserélni.

[108] n-et is növelhetjük. Persze, ha nagyon sokat kívánunk a géptől, a program elszállhat.

19.8.3. A Lagrange interpoláció

A 11.21 Megjegyzésben azt állítottuk, és egy feladatban azt kellett bebizonyítanunk, hogy $|x|$ Langrange interpolációja nem konvergál $|x|$-hez. Az alábbi MAPLE program ezt illusztrálja.

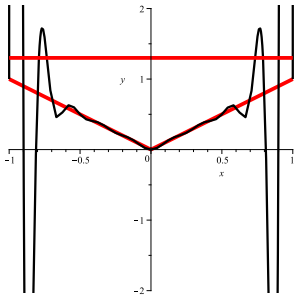
MAPLE MUNKALAP 4

```
> restart;                                              (a)
> n:=80;                                               (b)
> h:=1.3;                                              (c)
> with(CurveFitting);
> for i from 0 to n do
> x[i]:=-1+2*i/n:  y[i]:=abs(x[i]):                    (e) ertekadas
> end do:
> minta:=[seq([x[i],y[i]],i=0..n)];                    (m)
> g:=PolynomialInterpolation(minta,x);                 (LI)
> plot([abs(x),g,h],x=-1..1,y=-2..2,thickness=[5,3],color=black);   (r)
```

A programban, (b)-ben magadjuk n-et, a pontszámot. (c)-ben egy $h = 1.3$ vizszintes vonal magasságát rögzítjük, hogy majd lássuk, a görbe lényegesen 1 felett marad.

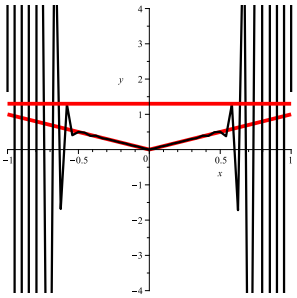
Egy i-ciklussal az x-abszolutérték adatait betesszük a minta nevű n-dimenziós blokkba (melynek elemei sikbeli pontok). (LI) kiszámolja a Lagrange Interpolációs polinomot, g-t. Végül (r)-ben kirajzoljuk az interpolációs görbét, $|x|$-et, és $h = 1.3$-at.

A program $n = 20, 40, 80$ osztópontra rendre a alábbi ábrákat produkálja:



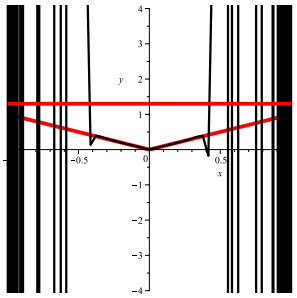
20 pont

19.22(a) ábra



40 pont

19.22(b) ábra



80 pont

19.22(c) ábra

Látható, hogy az interpolációs polinomok nem tartanak $|x|$-hez, de persze ez nem bizonyítás. Ugyanezt az ábrát kaphattuk volna numerikus instabilitás miatt is.

(Az end do helyett néhol **od** zárja a **do**-t.)

Nagyon leellenorizni

19.8.4. Differenciálegyenletek megoldása

A MAPLE differenciálegyenleteket is gyakran megold. Ez valóban kiemelkedő teljesítmény. A legegszerűbb differenciálegyenlet az integrálás, azaz, a primitív függvény megkeresése. Üssük be, hogy

```
> int(ln(x),x);
```

Erre a MAPLE integrálja a $\log x$ függvényt:

```
> x*ln(x)-x
```

$$x \ln(x) - x$$

A könyvben megoldottuk a rádioaktiv bomlás és a láncgörbe differenciálegyenleteit. A differenciálegyenletek elmélete nagyon mély és gyakran oldjuk meg az idevágó problémáinkat ,,ad hoc" módszerekkel.

Az alábbiakban megoldjuk a harmonikus rezgőmozgás másodrendű, állandó együtthatós differenciálegyenletét. Maga az egyenlet

$$f''(x) = -c \cdot f(x) \qquad \text{ahol } c \text{ konstans.} \tag{244}$$

A kérdés az, hogyan írjuk be.

```
> diff(f(x,y),x,y);
```

hatására a gép kiírja:

$$\frac{\partial^2}{\partial x \partial y} f(x,y)$$

Ezért beírjuk:

```
> ode:=diff(f(x),x,x)=-f(x);
```

azaz, $c = 1$-re akarjuk megoldani (244)-ot. Mire a gép ezt adja ki:

```
> ode := diff(f(x), x, x) = -f(x)
```

$$ode := \frac{d^2}{dx^2} f(x) = -f(x)$$

A következő sor

```
> dsolve(ode);
```

arra kéri meg a gépet, hogy oldja meg a fenti differenciálegyenletet[109] Kicsit barátságtalanabb alakban az alábbi (helyes) eredményt kapjuk:

$$f(x) = C_1 \sin x + C_2 \cos x.$$

30. FELADAT. Oldjunk meg néhány differenciálegyenletet a 227. oldalról.

Legyenek-e referenciaim?

References

[1] Garvan

[2] Simonovits Miklós honlapja: www.renyi.hu/˜miki

[109] ode = ordinary differential equation = közönséges, (azaz, egyváltozós) differenciálegyenlet.