Informatikatörténet

LGB_IN043_1

A BASIC programozási nyelv és a Dartmouth Időosztásos Rendszer története

Gráczol Benedek Péter

CX77QX

Győr, 2019/20/2.

Tartalom

Bevezetés

A dolgozatom központi témája a BASIC (Beginner's All-purpose Symbolic Instruction Code) programozási nyelv történetének bemutatása, annak megalkotásának előzményeitől kezdődően, Kemény János magyar származású matematikus aktív szerepvállalásán keresztül egészen a programnyelv technológiai hatásaiig, vagyis annak utóéletéig. Elsősorban azért választottam feladatul BASIC programozási nyelv témáját, hogy egyúttal rámutassak arra a mai is jelenlévő társadalmi és gazdasági igényre, hogy a programozás mesterségét érthető, könnyebben tanulható formában és minél szélesebb körben elérhetővé kell tenni. Ennek okát elsősorban a megnövekedett számítási és automatizálási szükségletekben kell keresni, de nagy népszerűségnek örvendenek azok a különféle platformokra (PC, okoseszközök) fejlesztett alkalmazások is, melyek használata a munkavégzéstől, a tanuláson át egészen a szórakozásig terjednek. Éppen ennek tükrében figyelemre méltó informatikatörténeti momentum az is, amikor 1964-ben Kemény János és Thomas Kurtz a Dartmouth főiskolán egy olyan általános célú (*All-purpose*) programnyelv kidolgozása mellett kötelezte el magát, amely elsősorban az egyetemi hallgatóság számítógépes tudásának gyarapítását, a jövőre nézve pedig mindennapos feladataiknak a könnyítését célozta meg. Azonban Thomas Kurtz saját bevallása szerint nem egy globális projektre gondolt akkor, amikor Kemény Jánossal a BASIC fejlesztésén kezdett el dolgozni, hanem ahogy az szavaiból kiderül, a programnyelvet csak a Darthmouth hallgatóinak és oktatóinak szánták.[1] Persze a BASIC, minthogy ez más információs technológiai vívmány esetében is tapasztalható volt, messzebbre jutott, mint azt megalkotói és felhasználói előzetesen elképzelték. Egy általános célú és az elődeihez képest könnyebben tanulható programnyelv megalkotása népszerű és gyorsan terjedő vállalkozásnak bizonyult a 70-es években és még a 80-as évek elején is. Idővel persze a növekedő és változó informatikai igények más programnyelvek megjelenéséhez vezettek, amelyek a 80-as években fokozatosan átvették a BASIC-t a szerepét, mind az oktatás, mind pedig a szoftverfejlesztés területén. A BASIC hatását ugyanakkor jól mutatja saját utóélete is, ami elsősorban azon programnyelvek előretörésében nyilvánul meg, melyeken mind szintaktikájában mind pedig felhasználási céljaiban a BASIC jellegzetességei tükröződnek. Itt elsősorban olyan a Microsoft által fejlesztett BASIC verziókat kell érteni, mint például a ma is használt Visual Basic.

A BASIC programozási nyelv fejlesztésével kapcsolatban, mindenképpen meg kell említeni azt a hardveres környezetet is, amiben megvalósult. Ezért elsőként egy másik korszakalkotó fejlesztésről, az általános célokra tervezett időosztásos, esetünkben a *Dartmouth Időosztásos Rendszerről* fogok írni. Ebben a fejezetben azt is látni fogjuk, hogy a programozás

megreformálása egybeesett a számítógéphasználat újraértelmezésével. A dolgozatomban természetesen szót ejtek mindezen technológiák megálmodójáról és megalkotójáról, a Magyarországról emigrált Kemény Jánosról is. A dolgozatom további részeiben először is a BASIC megjelenésének történeti előzményeit mutatom be, két programnyelvet érintve, majd ezt követően térek rá a BASIC nyelvre részletesebben. Beszámolok a nyelvről általánosságban, bemutatok néhány BASIC kódot a szemléltetés kedvéért, illetve befejezésül a hatásairól is ejtek néhány szót.

Kemény János útja Budapesttől Dartmouthig

Kemény János György Budapesten született 1926 május 31-én. Első iskolája a Rácz-féle elemi volt, ahol Neumann János is tanult. Középfokú tanulmányait egy híres budapesti iskolában a Berzsenyi Dániel Gimnáziumban végezte. Itt tanított többek között Benedek Marcel és ide járt iskolába például Károlyi Mihály és Tom Lantos.

Kemény János a Berzsenyi diákjaként 1939-ben
forrás: https://epa.oszk.hu/00300/00342/00043/mgy9305.html

Kemény János matematika iránti vonzalma már gyermekkorában megmutatkozott. Egy interjúban elmeséli, amikor kereskedő édesapja, Kemény Tibor könyvelője megmutatta neki a logaritmus táblázatot. Ekkor Kemény még csak 9 éves volt, de elmondása szerint már ennyi idős korában is nagy hatással volt rá ez az élmény.[2] Saját visszaemlékezése szerint mindig is vonzotta a matematika és nyilvánvaló volt számára, hogy egyszer matematikai pályára lép. Arra, hogy Kemény János hivatásszerűen foglalkozzon matematikával, legnagyobb hatással a Berzsenyi Gimnázium matematika tanára Bölcsházy Árpád volt. Élete során többször is felemlegette a fiatalon megtapasztalt magyar oktatási elveket, miszerint a tanárok a tanulókkal külön is foglalkoztak, amikor tanulmányi versenyre készültek. Tulajdonképpen az első kellemetlen tapasztalata az amerikai matematika oktatással kapcsolatban ebből a magyarországi benyomásból táplálkozik. *"A mi iskolánknak 5000 tanulója volt, de nem akadt egy matematika-tanár, aki támogatott volna a fölkészülésben. A harmadik helyezést értem el. Talán nem is rossz eredmény, ha figyelembe vesszük, hogy egy tanár sem biztatott vagy segített. Ez a nagy különbség New-York és Budapest között. - Jó matematikát csak Bölcsházy tanár úrtól tanultam három és fél évig. Ezután csak akkor, amikor a Princetoni Egyetemre kezdtem járni."* Kemény János az új világba való megérkezése után a *Georghe Washington High School* kezdett tanulni,

ahol az idézett eset is történt. Az Egyesült-Államokba való emigrációra a Magyarországon fokozódó zsidóellenesség miatt került sor, ami egyre nagyobb megpróbáltatásokat jelentett neki és családjának. Édesapja jól ismerte fel a helyzetet, amikor az Anschluss (Hitler 1938-ban bevonult Bécsbe, ezzel irányítása alá vonva Ausztriát) után kijelentette: „Ez a vég kezdete".[3] Kemény János a középiskola elvégzése után a Princetonra került, ahol ismét az elvárásainak megfelelő matematika oktatásban volt része. 1945-ben az amerikai állampolgárságot is megkapta, ami viszont katonai szolgálatra is kötelezte. A szolgálatát háborús hadszínterektől távol, Los Alamosban töltötte. Itt ismerkedett meg a későbbi Nobel-díjas fizikussal Richard Feynman-nal is, de itt találkozott más híres magyar származású magyar tudósokkal is, így például Neumann Jánossal, Wigner Jenővel és Teller Edével. A kivételes szellemi képességekkel megáldott magyarok találkozása ezen az új-mexikói katonai intézményben, nagy csodálkozást váltott ki a korabeli tudományos sajtóban, és egyenesen Marslakóknak tekintették őket. A Yankee folyóirat 1980 márciusi számában ezzel kapcsolatban idézi Kemény János reakcióját: „*John G. Kemeny enyhe marsbeli akcentussal hozzátette, hogy annyival könnyebb magyarul olvasni és írni, mint angolul, hogy a gyerekeknek sokkal több idejük marad a matematika tanulására.*"[3]

Kemény János feladata Los-Alamosban az volt, hogy elektromos kalkulátorokkal végezzen bonyolult számításokat a hadsereg munkáját támogatva. Ez a feladat lehetett az, ami megalapozta későbbi érdeklődését a számítógépek iránt. 1947-ben leszerelése után visszatért a Princetonra, ahol logikából diplomázott majd doktorált. 1949 és 1951 között a Prinecetoni Haditengerészeti Kutatóintézetben, 1951 és 1953 között pedig a filozófia tanszéken dolgozott. Alexander Fanelli Kemény Jánossal készített interjújában ennek hátterére is rákérdezett, amire Kemény a következő választ adta: „*Mindig is hobbim volt a filozófia. Végzős középiskolásként kezdtem filozófiát olvasni és komolyan lelkesedtem érte…Tiszta véletlen volt, hogy 51-ben állást kaptam. Kizárólag matematikusi állást kerestem, de a legjobb ajánlat a Princetonon egy filozófiai munkakör volt. Alig kellett 100 yardot megtennem és máris filozófussá váltam és talán még most is az lennék, ha nem kapok meghívást Dartmouthból*."[2] Dartmouthba, ahol az időosztásos számítógéprendszer és a BASIC programozási nyelv fejlesztését végezte, 27 évesen, Albert Einstein ajánlására került 1953-ban, akinek matematikus asszisztensként dolgozott. Erre Kemény János így emlékezett vissza: "*Einstein nem szorult másra fizikából. Matematikából segíteni kellett neki. Értette a matematikát, de nem volt korszerű matematika-tudása. Ehhez kellett az asszisztens. Hogy őszinte legyek, a matematika modern módszereihez jobban értettem.*"[3] Einsten-nel egyébként nem csak szakmai kapcsolatban állt, hanem nagyon

jó személyes viszony is volt közte és a világhírű fizikus között. „*Einstein volt a legkedvesebb, legaranyosabb ember, akivel életemben találkoztam.*”[3] – mondta erről Kemény János.

Dartmouth, Kemény János életének és munkásságának legfontosabb állomása volt, a nevéhez fűződő legfontosabb szakmai sikerei ugyanis ezen a 1769-es alapítású New Hampshire-i főiskolához köthetők. Az iskola szellemisége meglehetősen konzervatív volt, ezért nagyon komoly változásoknak számítottak azok az intézkedések, amelyeket később Kemény János, mint a főiskola rektora hozott meg. Kemény ugyanis, olyan nagy hatású fejlesztések után, mint amilyen a dolgozatom témáját képező **Dartmouth Időosztásos Rendszer** és a **BASIC** programozási nyelv voltak, nemcsak beírta magát az intézmény, sőt a világ informatika történetébe, de az iskola élére is került a 70-es években. Meggyőződése volt, hogy a számítógépes ismeretek széleskörű oktatásához, minden emberre szükség van. 200 év után megvalósította, hogy az amerikai őslakosok is beiratkozhassanak az egyetemre, illetve nagy ellenállással szembesülve, de kiharcolta, hogy nők is hallgathassák az előadásokat.[3]

Kemény János
forrás: https://web.archive.org/web/1960s.html

Kemény János nevéhez még számos eredmény köthető, ami főleg a Darthmouth főiskola informatikai fejlesztéseihez kapcsolódik, ezért szokás még az email szellemi atyjaként, vagy a számítógépes hálózatok ötletadójaként is számon tartani, A dolgozatomban az előző szakaszban említett két fejlesztéséről fogok bővebben szót ejteni. Ennek megfelelően elsőként a *Dartmouth Időosztásos Rendszert* fogom bemutatni, majd a BASIC programnyelvet ismertetem részletesebben.

A Dartmouth Időosztásos Rendszer

A DTSS (Dartmouth Time-Sharing System), vagy Dartmouth Időosztásos Rendszert Kemény János és Thomas Kurtz 1962-ben kezdte el fejleszteni. A munka befejezésének ideje 1964 május 1-jén hajnali 4 órakor történt meg, ami egyúttal a BASIC első programjának sikeres lefuttatását is jelentette:[4]

```
10 Print 2+2
20 End
```

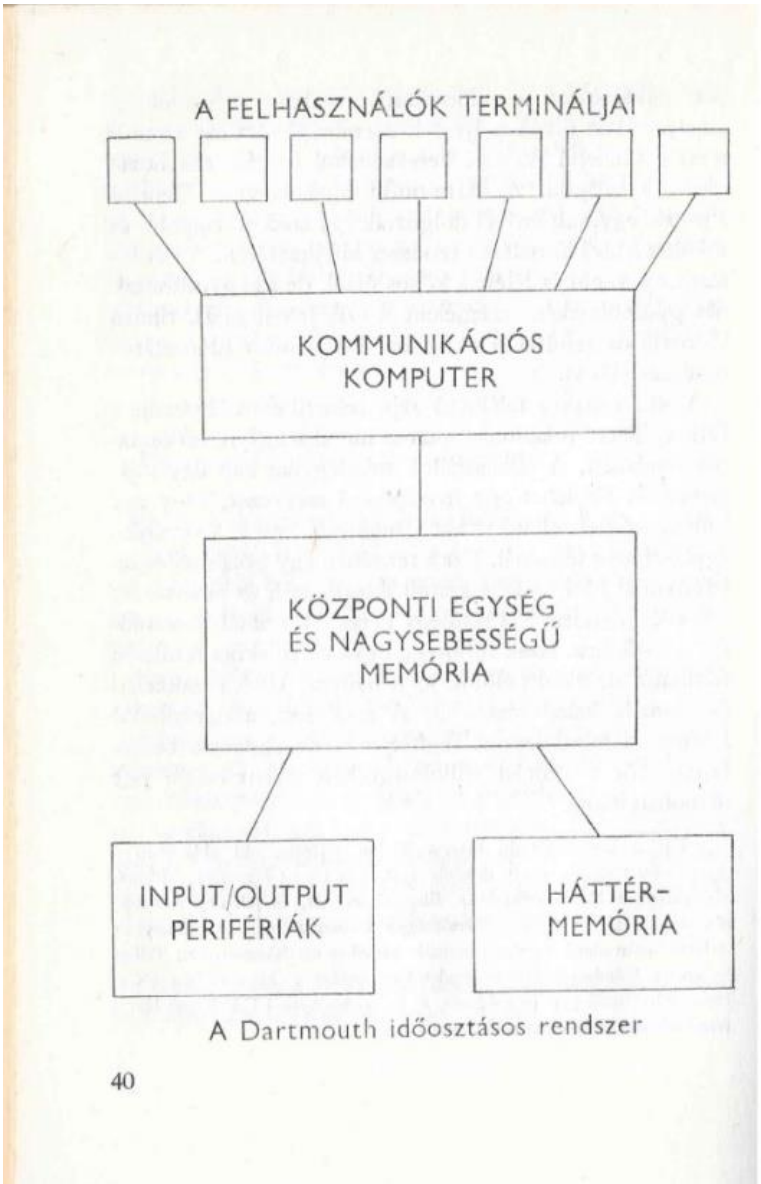
A DTSS létrehozására azért volt szükség, hogy a főiskola hallgatói könnyebben megismerjék a számítógépet. Kemény János meggyőződése volt ugyanis, hogy a jövő világában a számítógépek meghatározó szerepet fognak betölteni az emberek életében. Alighanem ebben igaza is lett. Ennek folyományaként, 1963-tól a dartmouthi tanrendbe is bekerült a számítógépek használata, mint kurzus. A bevezetéssel kapcsolatban a legnagyobb akadályt azonban nem egyfajta lemaradás, technológiai hátrány, hanem az olyan megoldások általános használata, mint a kötegelt (*batch*) utasításfeldolgozás jelentette. Akkoriban ugyanis, a számítógép felhasználók az értelmiségnek is csak egy szűkebb körét tették ki, így világos volt tehát, hogy a hallgatók körében más környezetet kell megvalósítani. „*Az a gondolat, hogy több száz hallgató bűvészmutatványokat végezzen az órarendjével, hogy naponta elmehessen a számítógépközpontba és megkapja a legújabb outputot, aztán hetekig vár arra, hogy végre rendesen működjön élete első programja, valóságos oktatási lidércnyomás lett volna.*”[5], ahogy maga Kemény János is rámutat az akkoriban megszokott számítógéphasználat nehézségeire. Ebben az esetben tehát nem feltétlenül a számítógép kapacitása vagy erőforrásai volt a fő probléma, hanem maga az adatfeldolgozási módszer. A kötegelt vagy más néven batch-feldolgozás hátránya leginkább abban mutatkozott meg, hogy a felhasználók nem tudták egyszerre többen önállóan igénybe venni a számítógépet. A kéréseket, inputokat egy-egy fizikai adathordozón, általában lyukkártyán adták be. Ezeket az operátorok egyesével összegyűjtötték majd kötegelve betáplálták a számítógépbe. Noha a számítógép egy átlagos programot másodpercek alatt feldolgozott, mégis hosszú órákat, esetenként napokat kellett várni arra, hogy a felhasználó megkapja saját munkájának kimenetét, outputját. Ráadásul csak tetézte a gondokat, hogy gyakran hibaüzeneteket kaptak vissza és javítani kellett a programot, amivel a folyamat előről kezdődött és tartott egészen addig, amíg a kívánt eredmény meg nem született. Kemény ezzel kapcsolatban a híres matematikus Norbert Wiener szavait idézi fel, amikor azt írja, mindez „*meglehetősen embertelen felhasználása volt az embereknek.*”[5] A fentiek tehát

rávilágítanak, hogy a szélesebb körű számítógép használat kialakítása érdekében valóban új megoldásra volt szükség az adatfeldolgozás terén.

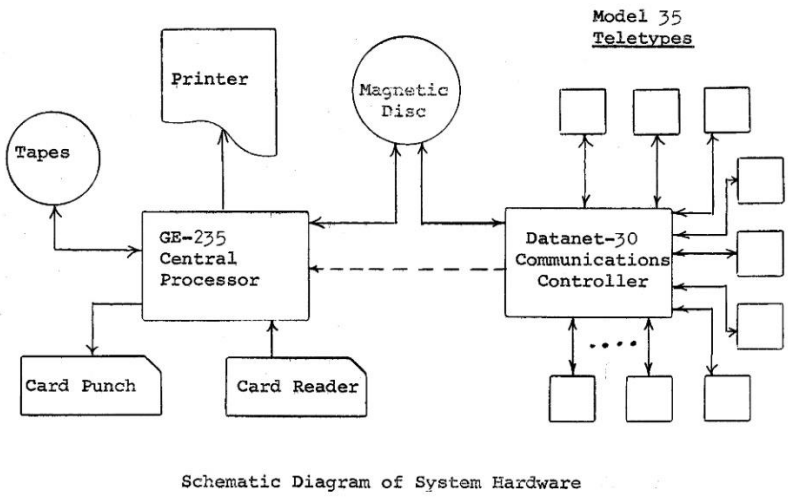
GE 225
forrás: https://web.archive.org/web/1960s.html

Kemény Jánosék a dartmouthi rendszer hardveres kialakítását a General Electric-vel (GE), az Egyesült Államok legnagyobb elektronikai vállalatával tervezték meg. A Dartmouth főiskolában a vállalat GE 225 típusú számítógépe került beállításra 1963 tavaszán. Majd ezt évekkel később további fejlesztések követték.[4] A Dartmouth Időosztásos Rendszert, és egyébként bármelyik időosztásos rendszert úgy kell elképzelni, hogy akár több felhasználó egy-egy saját terminálról használhatja a számítógépet. Bár a felhasználó részéről betáplált program futtatását a központi számítógép várakozó státuszba teheti (a központi számítógép processzora egy időben csak egyetlen feladattal foglalkozik), a felhasználó számára mégis úgy tűnik, hogy a gép csak az ő programjára koncentrál, így meglesz számára az a közvetlen kommunikációs élmény, ami a kötegelt megoldásánál nem volt lehetséges. A felhasználó, ahelyett, hogy egy teljes napot várna az eredményre, szinte másodperceken belül választ kap a számítógéptől. A központi számítógép egy felhasználó kérésével a másodperc tört részéig foglalkozik, majd pedig átlép egy másik felhasználó kérésére. Amennyiben ennyi idő alatt is megoldja a problémát, úgy a felhasználó szinte azonnali választ kap, amennyiben pedig nem, várakoznia kell, amíg egy másik időszeletben újra sorra kerül. Mindenesetre, mivel ez utóbbi esetben is csupán másodpercekről van szó, a felhasználónak az az illúziója támad, hogy valósidőben kap válaszokat a számítógéptől. Ennek az eljárásnak azonban volt egy további előnye is. A hallgatók ugyanis elkülönülve és szabadon gyakorolhatták a programozást, bátran követhettek el hibákat, ami nem csak azért volt kedvező számukra, mert nem kellett napokat várniuk a

javítási lehetőségre, hanem azért is mert megszabadultak a nyilvánosság terhétől. Ugyanez az előny az időosztásos rendszerek megjelenésével természetesen a tanárok számára is megadatott, talán esetükben ez még nagyobb jelentőséggel is bírt, hiszen működő, előzetesen tesztelt kódokon szemléltethették az oktatási anyagot. Az óra jelentős része ezzel felszabadult a bosszantó hibák javítása alól, és a lényegesebb programozói problémákra, algoritmusok megismertetésére koncentrálódhatott.



A DTSS sematikus ábrája 1.
forrás: John G. Kemeny: 40.o.[5]



Schematic Diagram of System Hardware

A DTSS sematikus ábrája 2.
forrás: https://en.wikipedia.org/wiki/Dartmouth_Time_Sharing_System

A fenti sematikus diagramok a Dartmouth Időosztásos Rendszer felépítését ábrázolják. Ahogy az ábrákon is látható, a felhasználók terminájai nem közvetlenül kapcsolódnak a központi egységhez, hanem egy kommunikációs számítógép közvetítésével. A kommunikációs egység szerepe, hogy a beviteli folyamatot feldolgozza és már kész kérést továbbítsa a központi gépnek. Erre a központi számítógép tehermentesítése érdekében van szükség. Mivel az egyes felhasználók programjainak betáplálása lényegében egyidőben történik, ezért az új megoldás lehetővé teszi, hogy a központi gépnek csak a már befejezett programokkal kelljen dolgoznia. A központi a gép a feladat elvégzése után, visszaküldi az eredményt a kommunikációs gépnek, ami kimenetként továbbítja azt a felhasználóhoz. A fenti ábrán (*A DTSS sematikus ábrája 1.*) látható a háttérmemória is, aminek legfontosabb szerepe az, hogy eltárolja a felhasználók programjait. A memóriának köszönhetően a felhasználók később is folytathatják, vagy továbbfejleszthetik programjaikat. Szintén az ábrán látható input/output perifériákon pedig elsősorban, az adathordozókat (lyukkártya, mágnesszalag) és a nyomtatók kezelőegységeit kell érteni.

Az időosztásos rendszer gyakorlati jelentősége tehát az volt, hogy nagyobb felhasználói csoportok, egész konkrétan főiskolai évfolyamok számára is lehetővé tegye a személyes kapcsolatot a számítógéppel, mindez Kemény János szavaival megfogalmazva: „*Az eredmény most is a komputer nagyfokú kihasználtsága, de az emberekre gyakorolt hatás teljesen más.*” [5] A kurzusok jellegét is alaposan megváltoztatta az újítás, hiszen a korábbi óráktól eltérően,

most valamennyi a kurzuson résztvevő hallgatónak lehetősége nyílt a terminálon keresztül közvetlenül használnia a számítógépet. Korábban maga a tanár futtatta le a programokat, miközben a hallgatók jegyzeteltek, vagy esetenként ők is kipróbálhatták a gépet, de várniuk kellett a sorukra, arról nem is beszélve, hogy az óra nagy része a hibák javításával telt.

Összességében tehát elmondható, hogy az időosztásos rendszer fejlesztése jól illeszkedik Kemény János azon törekvéséhez, hogy a számítógépet emberközelibb eszközzé formálja. Azonban ennél többről is szó volt. Kemény ugyanis egyfajta szimbióziskén tekintett az ember és a számítógép viszonyára és Neumann Jánoshoz hasonlóan mélyen hitt a számítógépek jövőjében. Könyvében, *Az ember és a számítógép*-ben erről részletesebben is beszámol. Kemény János személyében a számítógépekről alkotott jövőképhez, egy olyan személyiség is társult, melyből fakadóan meggyőződésesen vallotta, hogy az emberek szélesebb köreit alkalmassá lehet tenni a technológia elsajátítására. Erről az attitűdről tanúskodnak a matematika oktatásról vallott elvei is, mely szerint azok számára is közelebb kell hozni a matematikát, akik nem matematikusnak, sőt egyenesen bölcsész pályára készülnek. Kemény János sikerességét mutatja, hogy az ebben a szellemben szerkesztett *Introduction to Finite Mathematics (Bevezetés a véges matematikába)* című tankönyve 200 000 példányban fogyott el. Az egyik mottója a tanítással kapcsolatban pedig a következő volt: „*Tanítás az egyetlen gyógykezelés, ami használ nekem.*”[3] Kemény János munkásságának, szakmai sikereinek hátterében tehát mindenképpen meg kell látni azt a belső indíttatást, mely a fejlődés érdekében közelebb kívánta hozni az emberekhez a tudományt és a számítógépet.

Kemény János hallgatóinak társaságában
forrás: https://web.archive.org/web/1970s.html

Dartmouthi rendszerek az internet korszak előtt

Az időosztásos rendszereknek több előnye is volt a kötegelt feldolgozású rendszerekhez képest, az egyik tehát, hogy több felhasználó elméletileg egyidőben tudott dolgozni a programján, egy másik a munkák háttértárba mentése, de ilyen volt az is, aminek alapelve a mai napig meghatározza a számítógépes kommunikációs technológiát, ez pedig a hálózati megvalósítás. Bár egy internethez hasonló hálózatról itt még nem beszélhetünk, ahol a felhasználók interaktív kapcsolatba léphetnek egymással, mégis akkoriban forradalmi áttörésnek számított, hogy egy-egy terminál telefonhálózatra kötésével a távoli számítógéphasználat is valósággá vált. Dr. Myron Tribus egykori államtitkár, Xerox alelnök, hogy bemutassa az időosztásos rendszer egyik nagy előnyét, a skóciai Edinburgh-ben gépelt adatokat egy terminálon, amit aztán az Atlanti-óceán alatt húzódó interkontinentális telefonhálózaton keresztül továbbított a dartmouthi számítógépnek, Hanover városába, New Hampshire-be.[5] A 70-es évekre a dartmouthi rendszerhez már több száz terminál csatlakozott a telefonhálózat segítségével az USA észak-keleti területéről és Kanadából. Tulajdonképpen Montréál és Hanover összekapcsolásával vált nemzetközivé a rendszer. Konkrétabban itt a Dartmouthi Oktatási Hálózat (Dartmouth Educational Network) terjeszkedésről van szó, amire 1971-ben került sor. Ez a hálózat oktatási intézményeket, orvostudományi intézeteket, kormányzati kutató- és művészeti, valamint tudományos központokat foglalt magában. Megemlítendő még a 70-es években a főiskolán kidolgozott számítógépes könyvtári katalógus, ami 1987-ben már 850 000 könyvet és 30 000 folyóiratot katalogizált.

Itt csak néhány technológiai vívmányról esett szó, amit Dartmouthban fejlesztettek ki a 60-as évektől kezdődően egészen az internet megérkezésig, ami 1988-ban következett be, amikor is a főiskola az NSF (National Science Foundation (Nemzeti Tudományos Alap az Egyesült-Államokban) jóvoltából csatlakozhatott a princetoni Neumann János Központ számítógépéhez.[4]

A BASIC programozási nyelv

Mielőtt elkezdeném a BASIC programnyelvet tárgyaló fejezetet, röviden szót ejtek azokról a programnyelvekről is, amelyek időben és a logikailag is előzményének tekinthetők. Magát a BASIC-ről szóló fejezetet is négy részletben fogom bemutatni. Először általánosságban szót ejtek magáról a nyelvről, létrejöttének körülményeiről majd a szintaktikai jellemzőit reprezentálandó hozok kódpéldákat. Végezetül a nyelvet ért kritikákról, a BASIC utóéletéről, hatásairól számolok be röviden egy-egy alfejezetben.

Programozás a BASIC előtt

A BASIC előtti programnyelvek közül is azokat érdemes részletesebben megemlíteni, amelyek valamilyen módon hatással voltak rá. Ezek pedig az ALGOL (*Algorithmic Language*) illetve a Fortran (*Formula Translation*) programnyelvek. Mivel a BASIC fejlesztőinek alapvető célja az volt, hogy a programozást megkönnyítse a nem műszaki beállítottságú hallgatók számára is, ezért a fentebb említett programnyelvek éppen a speciális felhasználási céljaik miatt nem jöhettek szóba. A BASIC társfejlesztője Thomas Kurtz az egyszerűbb problémák programozását, így például a ciklusok létrehozását tartotta feleslegesen bonyolultnak, John McGeachie, aki pedig DTSS csapatának az egyik hallgatói tagja volt, a fenti programnyelvek nehézkes adaptációját nevezte meg legfőbb hátrányukként.[1] Összességében tehát elmondható, hogy szükség volt egy könnyebben tanulható, az egyszerűbb számítástechnikai problémák, feladatok megoldását végző programozási nyelvre is. Ezt a szükséget pedig elsőként a magyar származású matematikus, Kemény János ismerte fel.

Az időrendet betartva a Fortran programnyelvvel illik kezdeni, amelynek létrejötte John Backus nevéhez fűződik. Jelentőségét mutatja, hogy az 1957-ben megjelent nyelvet mind a mai napig alkalmazzák. A Fortran-t, ahogy arról elnevezése is árulkodik (*formula translation* = *képlet fordító*) elsősorban matematikai számítások, statisztikai elemzések, illetve mérnöki feladatok támogatására fejlesztették ki, tehát valódi általános célú megközelítésről itt még nem lehetett szó. Talán nem téves észrevétel az sem, hogy a Fortran szintaktikája is komplexebb, mint például a BASIC-é, nem is beszélve napjaink népszerű magas szintű programnyelveiről. Ezért elsősorban az az ellenvetés fogalmazódott meg a Fortran használatával szemben, hogy egyszerűbb, mondhatni hétköznapi feladatok megoldására nem tűnik a legoptimálisabb eszköznek.

A következőkben egy-egy ciklus példát mutatok be, amelyek persze csak felületesen képesek kifejezni a nyelvek közti szintaktikai különbségeket. A programozásban manapság is

gyakran használunk ciklusokat, iterációkat, különösképpen akkor, ha ugyanazt a műveletet többször is el szeretnénk végeztetni a számítógéppel. Ilyen műveletek például az egyszerű és összetett keresések, de ide tartoznak a számhalmazok, adatstruktúrák algoritmikus rendezései és még sokféle hasonló feladat. Most nézzünk egy-egy egyszerű példát a ciklus megvalósítására Fortran, C és természetesen BASIC nyelven. Itt megjegyzendő, hogy a C-t az eddig említett nyelvekhez képest időben jóval később dolgozták ki, említésére mégis azért kerülhet sor, mert szintaktikája jól mutatja azt a különbséget, amire itt a Fortran és BASIC vonatkozásában szó esik.

Ciklusok Fortran nyelven, a *do loop*, a *do while loop* típusban fordulnak elő, lássuk a *do loop*-ra egy egyszerű példát.

```
integer i
      do i = 0, 100, 1
         write(*,*) 'i =', i
```

forrás: https://www.tutorialspoint.com/fortran/

For-ciklus C-nyelven:

```
for(int i = 0; i < 100; i++)
     printf("i=%d", i);
```

For-ciklus BASIC nyelven:

```
FOR I = 0 TO 100 STEP 1
PRINT 'I=' I
NEXT
```

Az ALGOL nyelv a Fortran-hoz képest egy évvel később, 1958-ban jelent meg. Az ALGOL programnyelvnek három változata ismert, az 58, a 60 és a 68. Ezek közül az ALGOL 60 a legelterjedtebb. Az ALGOL fejlesztők alapvető célja az volt, hogy a Fortran bizonyos hibáit orvosolva egy letisztultabb és hatékonyabb programozási nyelv álljon a felhasználók rendelkezésére. A célok az ALGOL esetében is hasonlóak voltak, mint a Fortran-nál, azaz a bonyolultabb matematikai és tudományos számítások minél hatékonyabb elvégzése. Lényeges különbség a Fortran-hoz képest, legalábbis ami a programozás felhasználási területét illeti, az

ALGOL esetében sem tapasztalható. A szintaktikában azonban történtek változások, melyek közül például a pontosvessző használata az utasítások végén, illetve a blokk-struktúra (*begin - end*) megjelenése, igaz nem ebben a formában, de jócskán túlélte magát az ALGOL nyelvet. Valamilyen szintű strukturáltság tehát az ALGOL nyelv esetében is megjelent, aminek később a BASIC-et ért kritikák tükrében is jelentősége lesz. Az ALGOL-ban találkozhatunk először a *for-ciklussal*, ezt az alábbi példák közöl az elsőben láthatjuk:

- Syntax:
- `for var := <list_of_stuff> do statement`

- where <list_of_stuff> can have:
  - list of expressions
  - expression step expression until expression
  - expression while boolean_expression

```
for index := 1 step 2 until 50, 60, 70,
                80, index + 1 until 100 do
```

- (index = 1, 3, 5, 7, ..., 49, 60, 70, 80, 81, 82, ..., 100)

forrás: http://courses.cs.vt.edu/

```
// the main program (this is a comment)

BEGIN
FILE F (KIND=REMOTE);
EBCDIC ARRAY E [0:11];
REPLACE E BY "HELLO WORLD!";
WHILE TRUE DO
  BEGIN
  WRITE (F, *, E);
  END;
END.
```

forrás: http://groups.umd.umich.edu/

A BASIC bemutatása

Habár a Fortran és az ALGOL esetében is magas szintű - az utasítások végrehajtása nem közvetlenül gépi kóddal történik - programnyelvekről beszélünk, mégsem állíthatjuk, hogy velük beteljesült volna az az egyre inkább kialakuló elvárás, hogy a programozás szélesebb körben népszerű legyen. Valójában Kemény János volt az, aki elsőként megfogalmazta, hogy a programozást mindenki számára elérhetővé kell tenni és teljesíteni azt a célt, hogy az már ne csak a tudományos munkát végző szakemberek, valamint a mérnökök speciális munkaeszköze legyen. Kemény János meglátta a számítógéphasználatban rejlő lehetőségeket és tovább gondolta honfitársa Neumann János vízióit egy olyan világról, melyben a számítógépek meghatározó részei mindennapi életünknek. Kemény János egy szintén a gépi kódolástól már távoli, vagyis magas szintű, ugyanakkor a Fortran-nál és az ALGOL-nál könnyebben tanulható, emberközelibb nyelv kifejlesztését tűzte ki célul, ezért a következő elvárásokat támasztotta a BASIC-kel szemben:

1. A nyelvet a kezdő is könnyen megtanulhassa.
2. Sokoldalú nyelv legyen: bármilyen célra készülhessen program.
3. Magasszintű utasításai utólag tanulhatók, árát ne a kezdő fizesse, hanem a haladó.
4. A nyelv legyen interaktív használó és számítógép között.
5. Világos, érthető hibaüzeneteket adjon használóinak.
6. Kis programokra gyorsan válaszoljon.
7. Használható legyen a gép szerkezetének ismerete nélkül.
8. Védje a használót a computer operátorrendszerének gondjaitól.[3]

Mindezek a feltételek garantálták, hogy a BASIC esetében egy olyan programnyelv kerül az emberek kezébe, amely megnyitja számukra a programozás előtti kaput. A BASIC kialakítása tehát figyelembe vette a didaktikai szempontokat is, hiszen megszületésétől kezdve a programozás már nem volt többé egy szűk szakemberekből álló réteg kiváltsága és ebből következően hamar nagy népszerűségre tett szert. Természetesen ez hatott a számítógép gyártó cégek üzletpolitikájára is, és az első személyi számítógépek a BASIC-et választották programjaik működtetésére. Habár Kemény és a Kurtz a márkanevet levédték, azt szabadfelhasználásúvá tették, ezért aztán a számítógép gyártó cégek is igyekeztek kihasználni azt a lehetőséget, hogy saját üzleti céljaik szerint alakíthatják a nyelvet. Kezdetben a kis méretű

RAM és ROM korlátozta magát a programnyelvet is, vagyis bizonyos szempontból le kellett egyszerűsíteni azt. A probléma akkor jelentkezett, amikor a memóriakapacitás bővült, ugyanis a fejlesztők még a jobb hardveres feltételek mellett is ragaszkodtak egy szerényebb képességű BASIC-hez. Ez a helyzet azonban csak okot szolgáltatott más szakemberek kritikai megnyilvánulására, és abbéli céljaik megvalósítására, hogy aláássák a BASIC reputációját. Az egyik leggyakrabban hangoztatott kritika gyártók által módosított BASIC-kel szemben a nagyon sok ugrásos utasítás, az ún. GOTO használata. Ez az eljárás ugyanis szembement a 70-es években egyre nagyobb befolyást szerző strukturált programozás paradigmájával, ami a PASCAL, majd a C-nyelv megjelenésével vált meghatározóvá. A gyakorlati probléma a kódstruktúra hiányával az volt, hogy nehezítette a kód áttekinthetőségét és ezzel karbantarthatóságát. A szakemberek körében, ebben a helyzetben romlott a BASIC megítélése, miközben Kemény János azt hangoztatta, hogy a BASIC-kel eredetileg egy strukturált nyelvet alkottak, amit aztán a számítógépgyártók saját elképzeléseikhez igazítva butítottak le. Végül az említettek következményeként, 1984-ben Kemény és Kurtz gondozásában megjelent a BASIC egy PC-re fejlesztett változata, a True BASIC. Ezzel a változattal az volt a cél, hogy a strukturált programozás követelményeinek való megfelelés mellett egyúttal az egyszerűbb Mini BASIC-hez való alkalmazkodás is teljesüljön. Ez aztán egy sor gyártót arra ösztönzött, hogy erősebb BASIC-kel jöjjenek ki, így született meg például a Turbo BASIC, Quick BASIC, Visual BASIC.[3]

A BASIC első verziója 14 parancsot ismert, melyek egyaránt világos, természetes nyelvből származó kifejezések voltak. Ezek közül néhány a: PRINT, LET, IF-THEN, FOR-NEXT, GOTO, END. A PRINT paranccsal a kimenetre lehetett számokat vagy szöveget írni, a LET tulajdonképpen a változókat definiálta, az értékadást jelezte a feldolgozó egységnek, pl.: LET C = (A*2.5)+B;. Az IF-THEN elágazást a ma használatos programnyelveknél is jól ismerjük, igaz-hamis kérdések, problémák eldöntésénél fordul elő. A FOR és NEXT a ciklusok utasításait jelenti, a GOTO, amiről már volt szó, a programsorszám alapján viszi át/ugratja a vezérlést a megadott helyre. Az END parancs pedig nevéből adódóan a program végének jelzésére szolgált. Az INPUT később, 1966-ban a harmadik verzió megjelenésével került be a nyelvbe, ugyanakkor ez volt az a parancs, ami érdekessé tette a BASIC programozást. Az INPUT hiányában a BASIC mindössze általános matematikai problémák megoldására, illetve egyszerű szimulációk futtatására volt alkalmas. Az INPUT paranccsal a felhasználó alfanumerikus karakterek begépelésével különféle utasításokat adhatott a számítógépnek. Ez a parancsbővítés még számítógépes játékok programozására is alkalmassá tette a nyelvet.

Példák BASIC kódra

```
1 FOR N = 20 to 30
2 Print N, SQR(N)
3 Next N
4 End[5]
```

Az első itt bemutatott példakód, magától a kód fejlesztőjétől, Kemény Jánostól származik. Könyvében, az *Az ember és a számítógép*-ben, az alábbi kóddal szemlélteti milyen egyszerű szintaktika jellemzi a BASIC-et.

Itt egy egyszerű *for* ciklusról van tehát szó, mely sorra veszi a 20 és 30 közötti számok négyzetgyökét SQR(N), és kiíratja a szám mellé.

```
PDS> TYPE
FILE? MINTA.BAS

10 A = 22.3
20 B = 18.7
30 C = 28.1
40 S = A+B+C
50 X = S/3
60 PRINT 'ATLAGSZAMITAS'
70 PRINT
80 PRINT 'AZ ATLAG: X
90 PRINT
100 END

PDS>[6]
```

A következő kód azonban egy bevezető tankönyvi példa. Itt egy egyszerű átlagszámításról van szó, PRINT és END parancsok használatával. A felhasználói utasításokat a kódban szürke mezővel jelöltem. A terminálon a PDS> megjelenésével érhető el az operációs rendszer, ez után a TYPE beírásával lehet a jelezni a programnak, hogy a kimeneti állományt szeretnénk kiíratni a terminálra. A FILE? kérdést a rendszer teszi fel, a felhasználó pedig megadhatja, hogy pontosan melyik állományról van szó. Ennek rövidített formája egyébként a `PDS> TYPE MINTA.BAS`.

A következő példa már egy összetettebb BASIC programot mutat be. Itt szintén egy átlagszámítási feladatot láthatunk, ám ezúttal *for-ciklussal* történik a művelet végrehajtása. Az első ciklus az adatok bekérését végzi, míg a második ezen adatokból elvégzi az átlagszámítást és kiíratja a kimenetre, adott esetben egy képernyőre.

```
10    REM******************
20    REM*                 *
30    REM*      MERES       *
40    REM*                 *
50    REM******************
60    REM****ADATBEVITEL****
65    DIM T(100)
70    PRINT 'ADATBEVITEL'
80    PRINT
90    PRINT 'A MERESEK SZAMA:'
100   PRINT
110   INPUT M
200   REM* CIKLUS KEZDET *
210   FOR I=1 TO M
220         PRINT'AZ'I'.ADAT
KOVETKEZIK'
230         PRINT
240         INPUT T (I)
250         PRINT \ PRINT
260   NEXT I
270   REM* CIKLUS VEG*
300   REM***** ATLAGSZAMITAS *****
310   S=0
320   REM* CIKLUS KEZDET *
330   FOR I=1 TO M
340         S=S+T(I)
350   NEXT I
360   REM* CIKLUS VEG*
370   A=S\M
380   REM***** ATLAGKIIRAS *****
390   PRINT
400   PRINT 'A MERESEK SZAMA: 'M
410   PRINT
420   PRINT 'SZAMITOTT ATLAG: 'A[7]
```

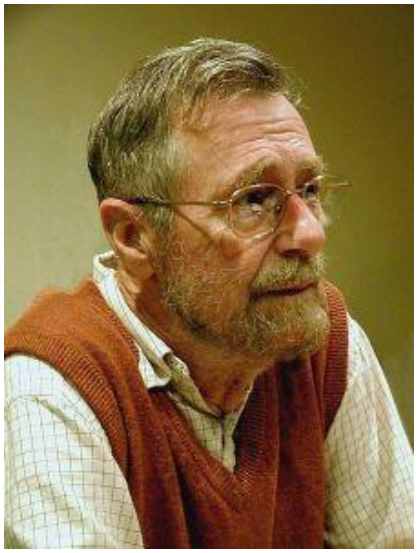
Az itt látható kódban a REM parancs utáni karaktereket a fordító figyelmen kívül hagyja, ez tehát az ún. „kommentelésre", megjegyzések hozzáfűzésre ad lehetőséget. A REM után gépelt szöveg tehát a kimeneten sem jelenik meg. A kódban a 65-ös sorszám mellett szereplő DIM T(100) egy legfeljebb 100 elemnek lefoglalt tömb deklarációja. A 110-es sorban megjelenik az INPUT parancs is, amely ebben az esetben a bekérendő adatok maximális számát kéri be a felhasználótól. Ez után tehát M darab számot adhatunk meg a programnak. Miután ez a ciklus végzett, a program elvégzi az átlagszámítást és mint ahogy az a 410 és a 420-as számú sorban is látható, kiíratja a mérések számát, M-t illetve a számított átlagot, A-t a terminálra. A programkódban szereplő önmagában álló PRINT-ek üres sorokat eredményeznek a kimeneten. Figyelemre méltó még a karakterláncok és a típusok külön kezelése a PRINT parancs mellett, ezeket a BASIC hasonlóan más nyelvekhez a karakterlánc idézőjelbe foglalásával különíti el egymástól.

```
100   GO TO 150
110   A=50
…
150   PRINT B[7]
```

A fentiekben egyszerű BASIC programokra hoztam példákat. A sokat kifogásolt GOTO parancs általában összetettebb programok esetében fordul elő, ahol a programvezérlés is gyakrabban változik. A bal oldalon csak a legegyszerűbb példája látható. A program a 100. sortól a 150.-ig ugrik, figyelmen kívül hagyva ezzel az A változót.

BASIC a kritikák tükrében

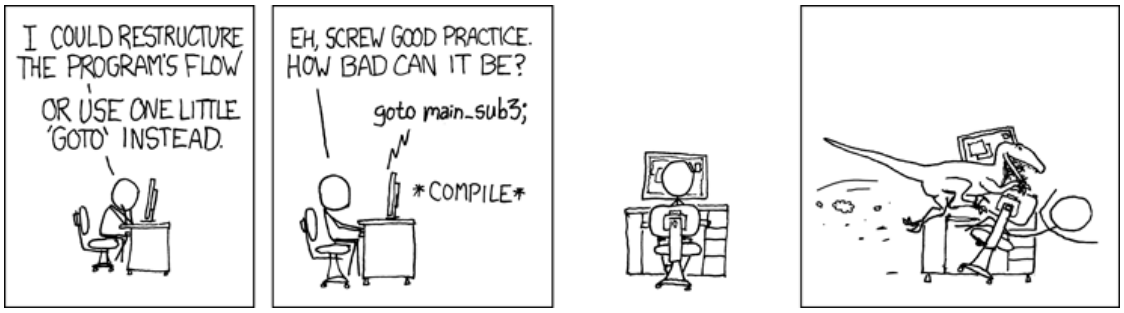
A BASIC-et ért leggyakrabban megfogalmazott vád, minthogy azt korábban említettem, jellemzően a nyelv programstruktúrát ignoráló viselkedésére vonatkozott. Védekezésképpen Kemény János gyakran hangoztatta, hogy az általuk fejlesztett BASIC nem azonos azokkal a verziókkal, amelyeket a vállalatok önkényesen átalakítottak. A strukturált nyelvek előretörésével a BASIC későbbi változatai, amelyeknek legismertebb példányai elsősorban a Microsoft gondozásában jelentek meg, nem igazán vették figyelembe a kódstruktúrára vonatkozó egyre határozottabb elvárásokat.



Edsger Wybe Dijkstra 2002-ben
forrás: https://time.com/69316/basic/

A BASIC egyik leghevesebb kritikusa a 70-es években Edsger Wybe Dijkstra holland származású matematikus, informatikus volt, akit a strukturált programozási paradigma első számú képviselőjeként tarthatunk számon. 1972-ben jelent meg erről szóló könyve *Structured Programming (Strukturált programozás)* címmel. Persze a GOTO és a programstruktúra szembenállása csak a vita felszíne volt, a problémák a BASIC nyelvvel kapcsolatban időnként nagyobb összefüggéseiben is előkerültek. „*Gyakorlatilag lehetetlen jó programozást tanítani azoknak a hallgatóknak, akik korábban a BASIC ártalmait megszenvedték, és mint potenciális programozók maradandó károsodást szenvedtek.*”[8] Dijkstra tulajdonképpen ilyen gúnyolódó hangnemben írta meg 1975-ben megjelent esszéjét, a „How Do We Tell Truths That Might Hurt?”-t, ami magyarul nagyjából annyit tesz: Hogyan mondjunk ki igazságokat, ami néha fáj? Az erős BASIC ellenességet némileg persze árnyalja az a tény, hogy a holland tudós hasonlóképpen nyilatkozott a Fortran, a PL/1 illetve a COBOL programnyelvekről is. A

Fortrant, gyermekbetegségként jellemezte, a PL/1-et halálos kórságnak, míg a COBOL-t egyenesen bűncselekménynek írta le esszéjében.[8]



Humoros kép a GOTO „veszélyeiről”
forrás: https://i.stack.imgur.com/mzvXU.png

Ugyanakkor, ahogy azt Kemény és Kurtz is sokszor elmondta a programnyelv védelmében, a BASIC-et sokkal inkább a küldetése alapján kell megítélni, ami pedig abban áll, hogy a programozást olyanok számára is élvezetessé tegye, akik korábban ódzkodtak ettől a tevékenységtől. A GOTO és a sorok számozása lehet, hogy nehezítette a program átláthatóságát és a frappáns megoldásokért lelkesedő szakemberek nem is nézték jó szemmel, mégis részesülhetett az alkotás örömében az a tanuló is, aki korábban még nem foglalkozott számítógépekkel. Ettől függetlenül persze a BASIC is alkalmas volt átlátható és elegánsabb kóddal szerkesztett programok megvalósítására, ám mivel a korszak általános tanulónyelve volt nagyon sok, kevésbé elegáns és talán még kevésbé hatékony kóddal is találkozhatott az ember. Nem csoda tehát, ha Dijkstra-ban és más szakemberekben is ellenérzéseket váltott ki egy-egy rosszul kinéző, gyengébb programkód. Ezzel szemben viszont Kurtz arra utalt, hogy a BASIC programokkal is problémákat oldanak meg, hasznos feladatokat implementálnak és a nyelvet ért támadásokban is inkább az irigységet vélte felfedezni.[1]

Thomas E. Kurtz a Teletype mellett
forrás: https://web.archive.org/web/1960s.html

A BASIC hatása

Kemény és Kurtz a DTSS-sel valami olyasmit alkotott, amit akár a személyi számítógépek előtörténetének is nevezhetünk. Mindenesetre a PC forradalom még egy évtizedet váratott magára. A 70-es évek ugyanis az első igazi PC-ék megjelenésével a BASIC programozás piacra törését is elhozta. Igaz ekkoriban már a BASIC különböző változatairól beszélhetünk, már nem arról, amit Kemény Jánosék Dartmouthban oktatási célokkal fejlesztettek ki a hallgatók számára, hanem inkább azokról, amik miatt a kritikákat is kapta a programnyelv.

A PC forradalmat az Egyesült Államokban az új-mexikói MITS vállalat Altair 8800 típusú számítógépe indította el. Paul Allen, aki Bill Gates iskolatársa, a későbbiekben pedig legközelebbi munkatársa volt, már az Altair megjelenése előtt saját BASIC-et fejlesztett. Allen számára akkor azonban még nem áll rendelkezésre olyan számítógép, amin a BASIC megfelelően lefutott volna. Az Altair megjelenésével ez az akadály is elhárult. Mivel Allen és Gates nem fért hozzá a gép rendszeréhez, egy szimulátort (*Digital Equiqment PDP-10*) használva írták meg a BASIC-et. Ebből a fontos munkakapcsolatból született meg végül a Micro-Soft, a világ legnagyobb tech-cégének az elődje.[1]

Bill Gates-ék végül az Altairrel közösen kezdték el forgalmazni saját BASIC verziójukat. 1977-ben a PC gyártás egy újabb hulláma érkezett el, olyan típusok megjelenésével, mint az Apple cég Apple II-es gépe, a Commodore PET 2001-es típusa, illetve a Radio Shack TRS-80-a. Mindegyik számítógépben volt valami közös, nevezetesen pedig az, hogy BASIC nyelven programozhatók voltak. Az Apple és a Radio Shack nem a Microsoft BASIC verziójával jöttek ki, de akkoriban általában elmondható volt, hogy a személyi számítógépeken Gates-ék szoftvere futott. Ilyen vállalatok voltak például az Atari és a Texas Instruments, melyek tehát MS BASIC-kel adták ki számítógépeiket. Persze nem csak Microsoft verziók voltak a piacon. Ilyen volt például a CBASIC, Gordon Eubanks fejlesztésében. 1985-ben azonban az eredeti Dartmouth BASIC fejlesztői Kemény János és Thomas Kurtz elégedetlenek voltak a különféle nyelvverziókkal és ennek hatására könyvet jelentettek meg *Back to BASIC*, magyarul *Vissza a BASIC-hez* címmel, melyben sérelmezték a Microsoft BASIC kimunkálatlanságát és logikátlanságát. Ekkor született meg a True BASIC.[1] Ez azonban már elkésett vállalkozás volt abban a tekintetben, hogy a 80-as évekre egyre inkább a PASCAL lett az iskolák általános oktató programnyelve az Egyesült-Államokban. A BASIC felett egyszerűen eljárt az idő és nem volt már olyan népszerű, mint a 70-es években.

Mindezekből persze levonhatnánk azt a következtetést, hogy a BASIC programnyelv szóra sem érdemes, hiszen egész egyszerűen eltűnt a történelem süllyesztőjében és manapság

sem akad olyan szoftverfejlesztő cég, amelyik használná ezt az egykoron népszerű nyelvet. Ugyanakkor persze a dolog ennél jóval árnyaltabb. Egyfelől magát a szándékot nézve, hogy a programozást népszerűsítse, másfelől pedig az olyan még ma is alkalmazott programnyelvek mint a Visual Basic, arra enged következtetni, hogy a BASIC bizonyos formában ma is jelen van az informatika világában A szándék felől közelítve, amelyet elsőként Kemény János tudott megfogalmazni, majd pedig ebből kiindulva képes volt megalkotni azt a programnyelvet, amellyel százezrek életébe hozta el a programozás élményét, a BASIC igen is történelmi és nagyhatású fejlesztés volt. Manapság is számos kampány, kezdeményezés szól a programozás népszerűsítése mellett, melyeket elsősorban a megnövekedett gazdasági igények generálnak. Azonban van itt még valami más is, ami túlmutat a programozás, és egyáltalán a számítógépek pragmatikus megismerésének céljánál, ez pedig nem más mint, amit Kemény János az ember és a számítógép szimbiózisának nevezett, amit Neumann János az emberiség jövőjeként megálmodott, amiben Raymond Kurzweil a szingularitást látta meg, mint az evolúció utáni valóságot. Azt olvashatjuk ki ezen kivételes elmék értekezéseiből, gondolataiból, hogy végsősoron a számítógépek, az informatika fejlődése nem pusztán egy technológiai fejlődés, hanem egyszersmind az ember fejlődése is.

Kemény János meglátta, hogy rendkívüli tétje van munkásságának és meglátta azt is, hogy merre visz az emberiség útja, és tudta, hogy mindezt úgy képes a leginkább szolgálni, ha a matematikát és a számítógépeket közelebb hozza az emberekhez. Sokan nem voltak képesek hasonlóan cselekedni és csak a maguk előtt tornyosuló problémákat látták, melyeket ráadásul a statisztikák szinte mindig megerősítettek. Ilyen előítéletek jellemezték például a matematika oktatás népszerűsítésének vagy a számítógéphasználat mindennapos rutinná formálásának gondolatát. Meglehet az előbbihez nem vezet királyi út, de az utóbbi idővel valóban mindennapjaink részévé vált és ennek a folyamatnak ráadásul koránt sincs vége.

Kemény János kétségtelenül nagyot alkotott, és nem csak amiatt mert kifejlesztett egy olyan programozási nyelvet, amit még azok is ismernek, akik nem éltek akkoriban, amikor a BASIC nagy népszerűségnek örvendett, hanem azért is, mert megmutatta a világnak, a technika úttörőinek, hogy mi az a magatartás, mi az az attitűd, amivel az emberiség nagyszerű céljait szolgálni kell.

Irodalom

[1] Harry McCracken (2014): Fifty Years of BASIC, the Programming Language That Made Computers Personal. https://time.com/69316/basic/ Letöltés időpontja: 2020. 05. 08.

[2] A. Alexander Fanelli(1984): John G. Kemeny – Interview. https://www.dartmouth.edu/~library/rauner/archives/oral_history/oh_interviews_pdf/KemenyInterview.pdf Letöltés időpontja: 2020. 05. 08.

[3] Marx György(1993): Kemény János. Fizikai Szemle 1993/5. 169.o https://epa.oszk.hu/00300/00342/00043/mgy9305.html Letöltés időpontja: 2020. 05. 08.

[4] Computing at Dartmouth. https://web.archive.org/web/20150425065704/http://www.dartmouth.edu/comp/about/archive/history/timeline/1960s.html Letöltés időpontja: 2020. 05. 08.

[5] John G. Kemény(1978): Az ember és a számítógép. Gondolat, Budapest.

[6] Dr. Kocsis András(1983):Programozás Basic Nyelven I. Számalk, Budapest.

[7] Dr. Kocsis András(1983):Programozás Basic Nyelven II. Számalk, Budapest.

[8] Edsger W. Dijkstra(1975): How do we tell truths that might hurt?. http://www.cs.virginia.edu/~evans/cs655/readings/ewd498.html Letöltés időpontja: 2020. 05. 08.