1963 **BASIC** = **B**eginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode är ett av de få högnivåspråk som inte är kompilerande utan *interpreterande*. Dvs källkoden tolkas rad för rad av datorns processor, utförs direkt och glöms bort sedan. Det uppstår ingen ny kod som lagras på hårddisken. Interpretering av källkod är alltid långsammare än exekveringen av redan kompilerad maskinkod. Däremot är interpretering snabbare än kompilering av källkod. I Basic finns inget kompileringssteg. Basic är, som namnet berättar, inriktat på att lära ut programmering för nybörjare. Därför har man hållit språket så enkelt som möjligt, så enkelt att man struntat i kontrollstrukturer som redan fanns i Algol och därmed lagt grunden för hoppsatser. Basic utvecklades ursprungligen av Dartmouth College i USA, men har sedan tagits över av Microsoft och integrerats som *QuickBasic* i DOS och Windows. På 90-talet har Microsoft lanserat vidareutvecklingen *Visual Basic* som blivit ett modernt och populärt utvecklingsverktyg. Den nyaste versionen heter *Visual Basic.NET* och är objektorienterad. I Visual Basic kan man även generera en exekverbar kod i efterhand genom att kompilera källkoden.

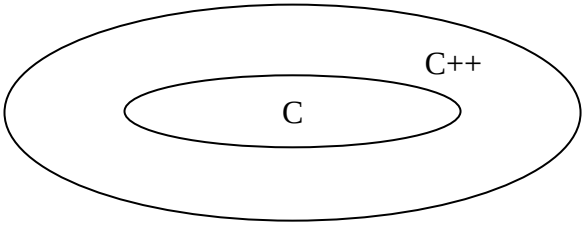
1971 **Pascal** är ingen förkortning för något utan har uppkallats efter *Blaise Pascal* som konstruerade räknemaskinen 1652. Pascal utvecklades av Niklaus Wirth på ETH (Eidgenössische Technische Hochschule) i Zürich. Tanken var att skapa ett kompilerande språk för att lära ut programmering för nybörjare genom att kombinera Basics enkelhet med Algols logiska strukturer och dess algoritmiska upplägg. På 80-talet utvecklade mjukvaruföretaget Borland *Turbo-Pascal* som blev en stor succé pga kompilatorns snabbhet och den *integrerade programutvecklingsmiljön (IDE)* som möjliggjorde kompilering, felsökning, editering och online hjälp i en och samma miljö. Idag marknadsför Borland Pascals objektorienterade vidareutveckling *Delphi*.

Från procedural (C) till objektorienterad programmering (C++)

70-80-talet **C++** är en direkt utvidgning och vidareutveckling av programmeringsspråket **C** som **1972** utvecklades av Dennis Ritchie på Bell Laboratories med syftet att skapa ett språk för programmering av operativsystemet *Unix*. I den bemärkelsen är C en biprodukt av Unix. Därför finns många logiska paralleller mellan C/C++ och Unix. Idag är inte bara Unix utan även andra operativsystem inkl. Windows skrivna i C/C++. Styrkan i C består av en kombination mellan enkelhet, strukturering och möjligheten att lätt kunna kommunicera med datorns hårdvara. C har bland de moderna språken den bästa förmågan att hantera och kontrollera hårdvaran, vilket favoriserar C som programspråk t.ex. för operativsystem. Den stora frihet som C erbjuder för hantering av bl.a. datorns primärminne med hjälp av *pekare*, kod som ger åtkomst till den fysiska adressen till data och på gott och ont tillåter manipulationer av minnesadresser genom *pekararitmetik*.

Det var dansken Bjarne Stroustrup som la grunden till vidareutvecklingen av C. Under 70-talet hade man nämligen konstaterat att *procedural programmering* (Algol, Pascal, C, …) inte längre tillgodosåg alla krav som stora komplexa program ställde med avseende på underhåll, förnyelse och ändringsbarhet. Ingen kunde sätta sig in i, ändra och vidareutveckla ett stort program om programmeraren hade lämnat företaget. Det innebar ett enormt slöseri med resurser. Dessutom utvecklades hårdvaruteknologin så snabbt att program som kunde köras på de allt mer avancerade datorerna blev allt större och mer komplexa, speciellt när det gällde grafiska tillämpningar. Mjukvaruteknologin utvecklades inte alls i samma takt. För att lösa alla dessa problem uppkom den nya programmeringsfilosofin *objektorienterad programmering* (OOP) som en vidareutveckling av den traditionella *procedurala programmeringen*.

1983 presenterade Bjarne Stroustrup programmeringsspråket C++. Han bibehöll hela C och la till de nya objektorienterade elementen, bl.a. klassbegreppet, som hade redan funnits t.ex. i *Simula*, ett norskt programmeringsspråk från 1967 som i sin tur var en direkt utbyggnad av *Algol (Algorithmic language)*. Simulas klasser hade ”glömts bort”. Den ovan beskrivna problematiken på 70-talet gjorde att man kom ihåg dem. Förhållandet mellan C och C++ illustrerar bäst den ”nya” filosofin tilläggskaraktär:



C är nämligen en delmängd av C++. Därför gäller all C-kod även i C++, men inte tvärtom. Dvs en C++ kompilator kan kompilera all kod skriven i C, men inte tvärtom. Så den som lär sig C++ lär sig automatiskt C. Delmängdrelationen mellan C och C++ är unik bland högnivåspråken.

C++ är ett kraftfullt och populärt programmeringsspråk, vars styrka ligger i textbaserade konsolapplikationer. Inte att C++ vore olämpligt för grafiska tillämpningar, bara att det är ganska jobbigt att skriva C++ kod för att åstadkomma grafik. En anledning är att, när C++ skapades, hade grafiska tillämpningar bara en begränsad spridning. Med utvecklingen av webbens grafiska miljö, med spridningen av Windows och grafiska användargränssnitt blev grafiken dominant. Idag har C++ fått en renässans med uppkomsten av *IoT* pga sin snabbhet och maskinnära egenskap.

90-talet **Unicode** är inget programmeringsspråk utan en internationell teckenkodningsstandard för utvidgning av ASCII-tabellen. Unicode är av betydelse för programmeringshistorien därför att den är en milstolpe mellan

textbaserade språk och sådana som inkluderar grafiska tillämpningar. Det är ingen slump att övergången av det textbaserade operativsystemet DOS till det fönsterbaserade Windows faller i samma period, se sid 44.

90-talet **Java**s uppkomst motiverades av en annan utveckling inom IT som man skulle kunna kalla den grafiska eller Webbrevolutionen. Urspungligen har Java utvecklats av *Sun Microsystems* som ett projekt för att skapa ett språk för programmering av hushållsmaskiner. Men detta projekt visade sig vara en bubbla som sprack som mycket annat inom IT. Webben, som revolutionerade IT, blev räddaren i nöden för Java. Men Java är inte bara grafik och webb. Sun satsade på att utveckla Java till ett universellt objektorienterat språk som var plattformsoberoende. Idag används Java bl.a. för webbapplikationer, t.ex. *Java Server Pages (JSP)*.

Sedan *Sun Microsystems* köpts upp av *Oracle*, är Java en Oracle-produkt. Oracle är en av världens ledande utvecklare av databashanterare. Java står inte i fokus av deras affärsverksamhet. Senaste tiden har Java tappat på popularitet inte minst pga sin lite krångliga kod jämfört med nyare utvecklingar som Python och C#.

90-talet **Python** skapades år 1989 av Guido van Rossum, en forskare på *National Research Institute for Mathematics and Computer Science* i Amsterdam och är en av de ovannämnda nyare utvecklingarna. Språket är *interpreterande* – liknande goda gamla BASIC – dessutom universellt. Python kan enkelt och gratis installeras på alla plattformar utan att man behöver bry sig om licenser. Koden är nästan självbeskrivande, ligger nära pseudokod och återspeglar algoritmen. I vissa avseenden är Python t.o.m. revolutionerande. Med små tekniska detaljer har man underlättat kodningen avsevärt. T.ex. har man avskaffat de obligatoriska symbolerna **{ }** för ett block. Det är inte längre nödvändigt att avsluta en sats med semikolon. De logiska indragningar som gör koden läsligare, har man lyft till obligatorisk syntax. Man är tvungen att följa god programmeringsstil. Variabler behöver inte explicit deklareras. Löpande kod och funktioner behöver inte nödvändigtvis skrivas i klasser. Språkets interpreterande karaktär gör det möjligt att på ett lekfullt sätt experimentera med kod. Pga dessa fördelar och sin enkla, smdidiga och kloka kodningsteknik har Python mer eller mindre konkurrerat bort Java och kan idag anses som världens mest populära programmeringsspråk åtminstone inom utbildning.

2000 **C#** har sina rötter i programspråken C, C++ och Java och är därmed byggt på det gamla, beprövade och välkända. Den allra första versionen av C# släpptes år 2000 av *Microsoft*. Man tog över allt som var bra och skrotade allt som var lite krångligt hos de andra språken. Men den viktigaste förnyelsen var att det nya språket integrerades i Microsofts .NET-miljö för att göra det utbytbart mot de andra språken inom .NET. En stor del av världens mjukvara utvecklas idag i C#.