Programmer's Swiss army knife

The Python computer language provides access to a universe of specialised tools

S ANANTHANARAYANAN

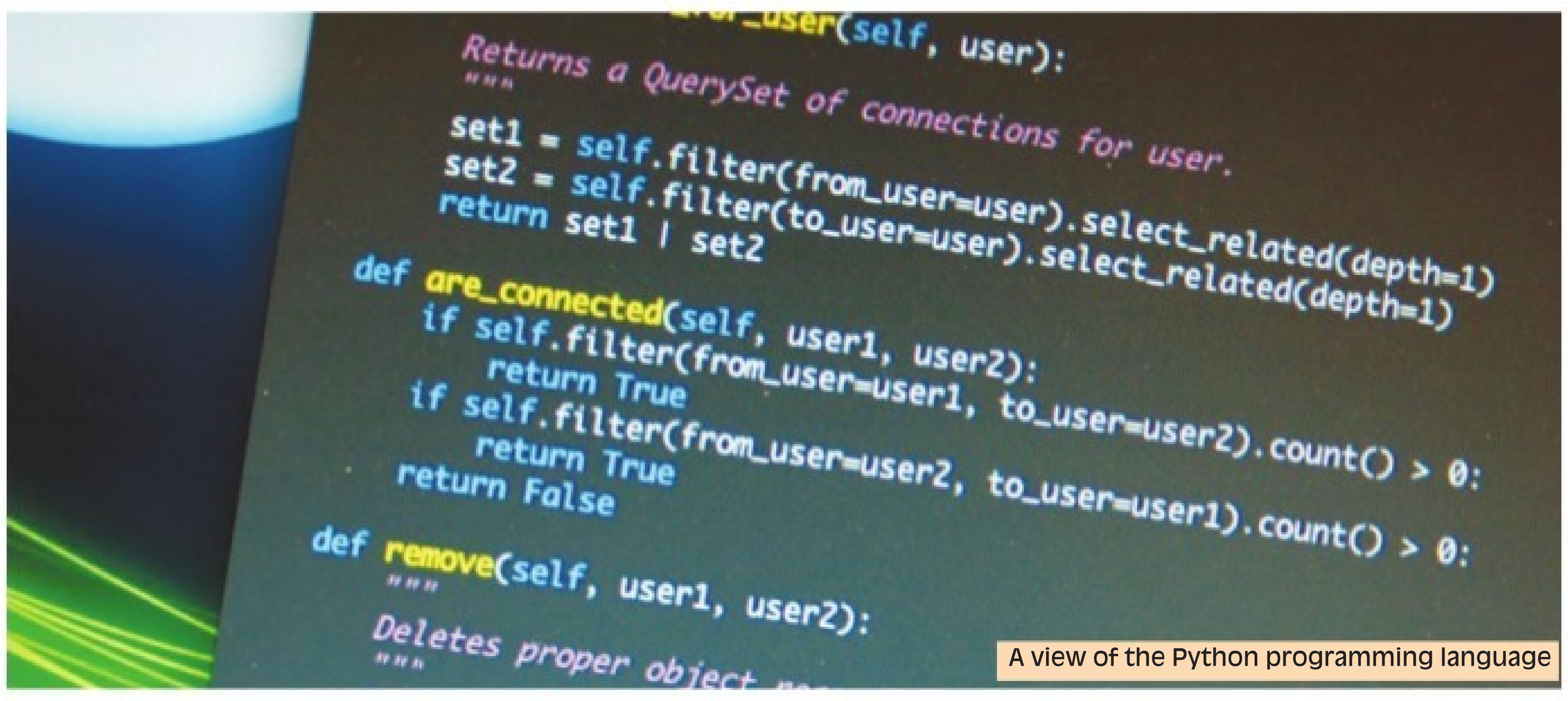
Software geeks who develop programming languages are not known for imagination in naming their creations. One of the earliest modern computer languages, developed in the 1960s, was called Basic Combined Programming Language or BCPL. The name came from its predecessor, CPL, for Cambridge Programming Language, which had been renamed Combined Programming Language.

Ken Thompson and Dennis Ritchie, pioneers of computer languages, modified BCPL to better suit its purposes and they named the new language, simply "B language". These computer languages were the first "higher level" interfaces between programmers and the raw circuitry of the machine, and specific languages were written for different machines. Ritchie then modified "B language" to create a versatile language that could easily be adapted to different machine architectures. As this language came after "B", Ritchie, in keeping with creativity in naming computer languages, named the new language "C".

Other popular computer languages, Fortran and Cobol, are application specific acronyms -- for Formula Translator and Common Business Oriented Language. The important development after C, the "object oriented" improvement, was C++, and this name deserves credit, as it is "one more than C" and it adds to "C" the C language code of "++" (meaning "increment by 1). And then, the next step after C++ is Java, which gets its name from the "endless cups of Java" that were consumed when the language was developed.

In this context, a computer language known as Python may stake a claim to be descriptive or evocative. But no, the language has nothing to do with serpents, its author, Guido van Rossum, was at a loss and he simply used the name of, "Monty Python's Flying Circus", a favourite BBC comedy show! But crises in finding names apart, languages have developed side by side with expanding applications and growing capability of computers.


Ken Thmpson Dennis Ritchie Guido van Rossum



A view of the Python programming language


The comedy show Monty Python's Flying Circus inspired the name

There has been tremendous growth in computer languages and great IT support is now available for data collection, computation, data analysis and data representation. The resources, however, are spread out and specialised, as opposed to being versatile. As applications need a host of technologies, programmers need the skills to work with many software packages. This is where the Python language steps in — it is a compact, user-friendly suite that brings together the tools and philosophy of different platforms. Python has hence found a wide field of application — "from testing microchips at Intel, to powering Instagram, to building video games", as one commentator says. In the university and the lab too, Python has become a handy computation and display aid for the scientist, engineer and student.

A special feature of Python code, which is what a programmer writes, or would reside in a device, is that it is economical, even brief. But the code is powerful, because it is a great store of domain-specific "libraries" that the short lines of code invoke. As an "open source" programming language, contributions in the form of code are added by programmers worldwide. Over the last 20 years that Python has been around, libraries and Python frameworks have grown, to provide eye-catching graphical displays for different fields of work, like earth sciences, astronomy and game and web development.

Getting a grip on Python, however, has not been easy, as it takes a degree of familiarity with several languages. The Princeton University Press has stepped in to supply the need with a new edition of *A Student's Guide to Python for Physical Modeling*. The book by Jesse M Kinder, assistant professor of physics at the Oregon Institute of Technology and Philip Nelson, professor of physics at the University of Pennsylvania, assumes little prior knowledge of programming and is addressed to undergraduates or any motivated reader.

The working of a computer is little more than storage of a "0" or "1" at a place in its "memory" and then manipulating the values, using just a few basic arithmetic operations that the electronics in the computer can carry out. The early languages were just methods to communicate with the computer to place values in memory locations, retrieve the values, carry out operations and then store results. The early programs, which were written in "machine language" were thus in symbols, which translated into electric signals that the computer could take and respond by storing, character by character, the symbols that followed, et al. And there was a sequence of symbols for each step in the computation process.

An advance was the "assembly level" language, which used "human words", like "read" or "write" to take the place of machine language code. There still was, however, one assembly level command for each machine language command, although there were the "macros", which combined more than one of some assembly level commands, to make things easier. The more versatile, "higher level" languages were in fact programs like these, but they defined a grammar.

These languages defined classes of values, "integer" or "character" (which automated the storage space to be reserved for instance). This enabled the user to use easy syntax of values and commands, while the task of writing out the assembler or machine code was done by a "compiler" program. Much of the detailed, machine-specific housekeeping, managing the memory, and later the display, was also taken over by another program, the "operating system". Programming languages were then created to speak to the operating system and the user's job was greatly simplified.

Languages then grew to have the capability of creating user defined "functions", or new "macros", which could be readily invoked to carry out a complex computation or tabulation. Similar functions were built into the language, to take control of the display, to create text boxes or spaces where a user could be invited to enter data, to make connection over a network and so on. As the use of IT grew and large programs were written, there arose the need for the language to help create error-free programs and to make use of functions that were written by other programmers and available in a separate folder, or even over the Internet.

There was a progression, with the first programming languages being "procedural" and then "structured". But with growing complexity, there arose the need for greater abstraction and to view procedures as interactions between isolated and protected entities known as "objects". This was a powerful innovation, where objects and generations of objects, had specific properties and means of communication. The concept made for libraries of objects of different kinds, built in with properties and methods, optimised and tested, which could be invoked and adapted, for extremely fast software development.

The Python language incorporates these features and the programmer gets access to a universe of specialised tools, which he or she can adapt. And the student's guide from Princeton University Press, systematically and rapidly, hand-holds a novice through the basics — mathematical computation and generating 3D models and displays.

The writer can be contacted at response@simplescience.in

Taking down megafauna

Here's how our ancestors hunted the giant sloth but the reasons for doing so are yet unclear

Rearing on its hind legs, the giant ground sloth would have been a formidable prey for anyone, let alone humans without modern weapons. Tightly muscled, angry and swinging its forelegs tipped with wolverine-like claws, it would have been able to defend itself effectively. Our ancestors used misdirection to gain the upper hand in close-quarter combat with this deadly creature.

What is perhaps even more remarkable is that we can read this story from the 10,000-year-old footprints that these combatants left behind, as revealed by our new research published in *Science Advances*. Numerous large animals such as the giant ground sloth — so-called megafauna — became extinct at the end of the Ice Age. We don't know if hunting was the cause, but the new footprint evidence tells us that human hunters clearly did tackle such fearsome animals and the way in which they did it.

These footprints were found at White Sands National Monument in New Mexico, US, in an area used by the military. The White Sands Missile Range, located close to the Trinity nuclear site, is famous as the birthplace of the US space programme, of Ronald Reagan's Star Wars initiative and of countless missile tests. It is now a place where long-range rather than close-quarter combat is fine-tuned.

It is a beautiful place, home to a huge salt playa (dry lake) known as Alkali Flat and the world's largest gypsum dune field, made famous by numerous films including *Transformers* and *The Book of Eli*. At the height of the Ice Age it was home to a large lake (Lake Otero).

As the climate warmed, the lake shrank and its bed was eroded by the wind to create the dunes and leave salt flats that periodically pooled water. The Ice Age megafauna left tracks on these flats, as did the humans that hunted them. The tracks are remarkable in that they are only a few centimetres beneath the surface and yet have been preserved for over 10,000 years.

There are tracks of extinct giant ground sloths, mastodons, mammoths, camels and dire wolves. These tracks are colloquially known as "ghost tracks" as they are only visible at the surface during specific weather conditions, when the salt crusts are not too thick and the ground not too wet. Careful excavation is possible in the right conditions and reveals some amazing features.

Perhaps the most remarkable of these is a series of human tracks that we found within the sloth prints. In our paper, produced with a large number of colleagues, we suggest that the humans stepped into the sloth prints as they stalked them for the kill. We have also identified large "flailing circles" that record the sloth rising up on its hind legs and swinging its fore legs, presumably in a defensive, sweeping motion to keep the hunters at bay. As it overbalanced, it put its knuckles and claws down to steady itself.

These circles are always accompanied by human tracks. Over a wide area, we see that where there are no human tracks, the sloths walk in straight lines. Where human track are present, the sloth trackways show sudden changes in direction, suggesting the sloth was trying to evade its hunters.

Piecing together the puzzle, we can see how sloths were kept on the flat playa by a horde of people who left tracks along its edge. The animals was then distracted by one stalking hunter, while another crept forward and tried to strike the killing blow. It is a story of life and death, written in mud.

What would convince our ancestors to engage in such a deadly game? Surely the bigger the prey, the greater the risk? Maybe it was because a big kill could fill many stomachs without waste or maybe it was pure human bravado.


Plaster cast giant sloth footprints

At this time at the end of the last Ice Age, the Americas were being colonised by humans spreading out over the prairie plains. It was also a time of animal extinctions. Many palaeontologists favour the argument that human over-hunting drove this wave of extinction and for some it has become an emblem of early human impact on the environment. Others argue that climate change was the true cause and our species is innocent.

It is a giant crime scene in which footprints now play a part. Our data confirms that human hunters were attacking megafauna and were practiced at it. Unfortunately, it doesn't cast light on the impact of that hunting. Whether humans were the ultimate or immediate cause of the extinction is still not clear. There are many variables including rapid environmental change to be considered. But what is clear from tracks at White Sands is that humans were then, as now, "apex predators" at the top of the food chain.

Written by Matthew Robert Bennett, professor of environmental and geographical sciences; Katie Thompson, research associate, and Sally Christine Reynolds, senior lecturer in hominin palaeoecology at Bournemouth University, UK

The independent

PLUS POINTS

Latest discovery

Hidden writing on fragments of the Dead Sea Scrolls has been discovered by researchers using advanced imaging technology originally developed for Nasa.

The pieces of manuscript, many of which appear blank to the naked eye, are being analysed for the first time since they were unearthed by archaeologists in the 1950s.

One fragment had letters written in an ancient Hebrew script which could belong to an unknown text. Others have been matched to the books of Deuteronomy and Leviticus, a section of the Great Psalms Scroll containing Psalm 147, and a text in the Temple Scroll giving instructions for how to conduct services.

The fragments, roughly around 1.5cm by 1cm in size, were presented during an international conference held in Jerusalem to mark the 70th anniversary of the discovery of the scrolls. "Usually scholars are dealing with Dead Sea Scrolls materials after they've been cleaned and treated. These fragments were never cleaned and treated," said Oren Ableman, a researcher at the Dead Sea Scrolls Unit of the Israel Antiquities Authority.

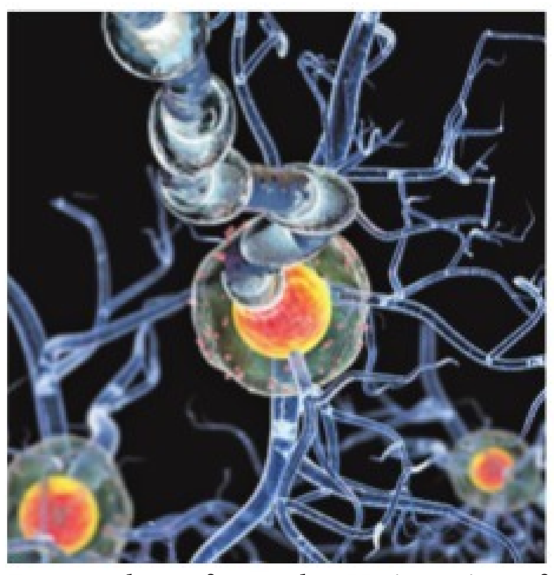
Using an infrared microscope connected to a computer, he sifted through a box of fragments recovered from Cave 11 in 1956. The fragments were then photographed using multispectral imaging as part of the decade-long project to digitise the Dead Sea Scrolls.

Ableman believes one piece with paleo-Hebrew writing dates back to the late Second Temple period. "What was exciting about this particular fragment is that I could tell that the handwriting was not identical to other fragments of this type of script," he said.

The Dead Sea Scrolls were first discovered by Bedouin shepherds in 1946 in caves at Wadi Qumran in the West Bank. In February last year archaeologists discovered a new cave containing pottery, broken scroll storage jars and one blank parchment.

The independent

Understanding gait



Researchers from the University of Sheffield and Sheffield Teaching Hospitals NHS Foundation Trust have developed an algorithm that, when paired with wearable sensors, provides more informative and effective monitoring of the way multiple sclerosis patients walk in real life. The study called "Free-living and laboratory gait characteristics in patients with multiple sclerosis" was recently published in the journal, *PLOS ONE*.

Assessing the way a person walks (gait) is often used as an indicator in the early stages of MS — a chronic autoimmune inflammatory disease of the central nervous system. Mobility problems affect 75 to 90 per cent of people with MS.

Up until now gait analysis has only been carried out in laboratories. Doctors at Sheffield Teaching Hospitals approached researchers at the University of Sheffield and asked them to help find a way to measure how patients walk in "real life" conditions.

Dr Claudia Mazzà, a researcher based at the Insigneo Institute for in silico Medicine at the University of Sheffield, said, "We started off by checking that our portable sensor was accurate, comfortable and able to give the same results as a lab-based sensor. We then developed an algorithm (computer program) specific to the patient's condition (in this case MS) which processed the measurements taken from this sensor. Although this is a small study, the results are encouraging and it gives us enough information to progress to a large scale clinical trial."

Dr Sivaraman Nair, consultant neurologist at Sheffield Teaching Hospitals, said, "Assessing the changes in the way patients with MS walk is key to understanding the progression of disability. The potential applications of this research are not just limited to MS but could be used for other conditions that could benefit from monitoring gait, such as Parkinson's disease."