TRƯỜNG THPT XUÂN LỘC

TIN HỌC 11

Năm học 2021-2022

(Lưu hành nội bộ)

MỤC LỤC

Trang

CHƯƠNG I.
MỘT SỐ KHÁI NIỆM VỀ LẬP TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH

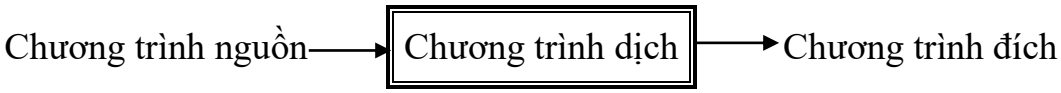
§1. KHÁI NIỆM LẬP TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH

Như đã biết, mọi bài toán có thuật toán đều có thể giải được trên máy tính điện tử. Khi giải bài toán trên máy tính điện tử, sau các bước xác định bài toán và xây dựng hoặc lựa chọn thuật toán khả thi là bước lập trình.

Lập trình là sử dụng cấu trúc dữ liệu và các câu lệnh của ngôn ngữ lập trình cụ thể để mô tả dữ liệu và diễn đạt các thao tác của thuật toán. Chương trình viết bằng ngôn ngữ lập trình bậc cao nói chung không phụ thuộc vào máy, nghĩa là một chương trình có thể thực hiện trên nhiều máy. Chương trình viết bằng ngôn ngữ máy có thể được nạp trực tiếp vào bộ nhớ và thực hiện ngay còn chương trình viết bằng ngôn ngữ lập trình bậc cao phải được chuyển đổi thành chương trình trên ngôn ngữ máy mới có thể thực hiện được.

Chương trình đặc biệt có chức năng chuyển đổi chương trình được viết bằng ngôn ngữ lập trình bậc cao thành chương trình thực hiện được trên máy tính cụ thể được gọi là *chương trình dịch*.

Chương trình dịch nhận đầu vào là chương trình viết bằng ngôn ngữ lập trình bậc cao (chương trình nguồn) thực hiện chuyển đổi sang ngôn ngữ máy (chương trình đích).



Xét ví dụ, bạn chỉ biết tiếng Việt nhưng cần giới thiệu về trường của mình cho đoàn khách đến từ nước Mĩ, chỉ biết tiếng Anh. Có hai cách để bạn thực hiện điều này.

Cách thứ nhất: Bạn nói bằng tiếng Việt và người phiên dịch giúp bạn dịch sang tiếng Anh. Sau mỗi câu hoặc một vài câu giới thiệu trọn một ý, người phiên dịch dịch sang tiếng Anh cho đoàn khách. Sau đó, bạn lại giới thiệu tiếp và người phiên dịch lại dịch tiếp. Việc giới thiệu của bạn và việc dịch của người phiên dịch luân phiên cho đến khi bạn kết thúc nội dung giới thiệu của mình. Cách dịch trực tiếp như vậy được gọi là *thông dịch*.

Cách thứ hai: Bạn soạn nội dung giới thiệu của mình ra giấy, người phiên dịch dịch toàn bộ nội dung đó sang tiếng Anh rồi đọc hoặc trao văn bản đã dịch cho đoàn khách đọc. Như vậy, việc dịch được thực hiện sau khi nội dung giới thiệu đã hoàn tất. Hai công việc đó được thực hiện trong hai khoảng thời gian độc lập, tách biệt nhau. Cách dịch như vậy được gọi là *biên dịch*.

Sau khi kết thúc, với cách thứ nhất không có một văn bản nào để lưu trữ, còn với cách thứ hai có hai bản giới thiệu bằng tiếng Việt và bằng tiếng Anh có thể lưu trữ để dùng lại về sau.

Tương tự như vậy, chương trình dịch có hai loại là *thông dịch* và *biên dịch*.

a. Thông dịch

Thông dịch được thực hiện bằng cách lặp lại dãy các bước sau:

① Kiểm tra tính đúng đắn của câu lệnh tiếp theo trong chương trình nguồn;

② Chuyển đổi câu lệnh đó thành một hay nhiều câu lệnh tương ứng trong ngôn ngữ máy;

③ Thực hiện các câu lệnh vừa chuyển đổi được.

Như vậy, quá trình dịch và thực hiện các câu lệnh là luân phiên. Các chương trình thông dịch lần lượt dịch và thực hiện từng câu lệnh. Loại chương trình dịch này đặc biệt thích hợp cho môi trường đối thoại giữa người và hệ thống. Tuy nhiên, một câu lệnh nào đó phải thực hiện bao nhiêu lần thì nó phải được dịch bấy nhiêu lần.

Các ngôn ngữ khai thác hệ quản trị cơ sở dữ liệu, ngôn ngữ đối thoại với hệ điều hành,... đều sử dụng trình thông dịch.

b. Biên dịch

Biên dịch được thực hiện qua hai bước:

① Duyệt, kiểm tra, phát hiện lỗi, kiểm tra tính đúng đắn của các câu lệnh trong chương trình nguồn;

② Dịch toàn bộ chương trình nguồn thành một chương trình đích có thể thực hiện trên máy và có thể lưu trữ để sử dụng lại khi cần thiết.

Như vậy, trong thông dịch, không có chương trình đích để lưu trữ, trong biên dịch cả chương trình nguồn và chương trình đích có thể lưu trữ lại để sử dụng về sau.

Thông thường, cùng với chương trình dịch còn có một số dịch vụ liên quan như biên soạn, lưu trữ, tìm kiếm, cho biết các kết quả trung gian,... Toàn bộ các dịch vụ trên tạo thành một môi trường làm việc trên một ngôn ngữ lập trình cụ thể. Ví dụ, Turbo Pascal 7.0, Free Pascal 1.2, Visual Pascal 2.1,... trên ngôn ngữ Pascal; Turbo C++, Visual C++,... trên ngôn ngữ C++; ngôn ngữ Python.

Các môi trường lập trình khác nhau ở những dịch vụ mà nó cung cấp, đặc biệt là các dịch vụ nâng cấp, tăng cường các khả năng mới cho ngôn ngữ lập trình.

§2. CÁC THÀNH PHẦN CỦA NGÔN NGỮ LẬP TRÌNH

1. Các thành phần cơ bản

Mỗi ngôn ngữ lập trình thường có ba thành phần cơ bản là *bảng chữ cái, cú pháp* và *ngữ nghĩa.*

a. Bảng chữ cái là tập các kí tự được dùng để viết chương trình. Không được phép dùng bất kì kí tự nào ngoài các kí tự quy định trong bảng chữ cái.

Bảng chữ cái bao gồm các kí tự:

- Các chữ cái thường và các chữ cái in hoa của bảng chữ cái tiếng Anh:

  a b c d e f g h i j k l m n o p q r s t u v w x y z
  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- 10 chữ số thập phân Ả Rập: 0 1 2 3 4 5 6 7 8 9
- Các kí tự đặc biệt:

+	-	*	/	=		<	>	[	]	.	\	,
;	#	^	$	@	&	(	)	{	}	:	!	' "
dấu cách (mã ASCII 32)												_

b. Cú pháp là bộ quy tắc để viết chương trình. Dựa vào chúng, người lập trình và chương trình dịch biết được tổ hợp nào của các kí tự trong bảng chữ cái là hợp lệ và tổ hợp nào là không hợp lệ. Nhờ đó, có thể mô tả chính xác thuật toán để máy thực hiện.

c. Ngữ nghĩa xác định ý nghĩa thao tác cần phải thực hiện, ứng với tổ hợp kí tự dựa vào ngữ cảnh của nó.

Ví dụ: Phần lớn các ngôn ngữ lập trình đều sử dụng dấu cộng (+) để chỉ phép cộng. Xét các biểu thức:

A + B (1)

I + J (2)

Giả thiết *A, B* là các đại lượng nhận giá trị thực và *I, J* là các đại lượng nhận giá trị nguyên. Khi đó dấu "+" trong biểu thức (1) được hiểu là cộng hai số thực, dấu "+" trong biểu thức (2) được hiểu là cộng hai số nguyên. Như vậy, ngữ nghĩa dấu "+" trong hai ngữ cảnh khác nhau là khác nhau.

Tóm lại, cú pháp cho biết cách viết một chương trình hợp lệ, còn ngữ nghĩa xác định ý nghĩa của các tổ hợp kí tự trong chương trình.

Các lỗi cú pháp được chương trình dịch phát hiện và thông báo cho người lập trình biết. Chỉ có các chương trình không còn lỗi cú pháp mới có thể được dịch sang ngôn ngữ máy.

Các lỗi ngữ nghĩa khó phát hiện hơn. Phần lớn các lỗi ngữ nghĩa chỉ được phát hiện khi thực hiện chương trình trên dữ liệu cụ thể.

2. Một số khái niệm

a. Tên

Mọi đối tượng trong chương trình đều phải được đặt tên theo quy tắc của ngôn ngữ lập trình và từng chương trình dịch cụ thể.

Trong Python tên là một dãy liên tiếp bao gồm chữ số, chữ cái hoặc dấu gạch dưới và bắt đầu bằng chữ cái hoặc dấu gạch dưới.

Ví dụ: trong ngôn ngữ Python:

- Các tên đúng: a
  r21
  p21_c
  _45
- Các tên sai:
  a bc (chứa kí tự trắng)
  6pq (bắt đầu bằng chữ số)
  x#y (chứa kí tự "#" không hợp lệ)

Lưu ý: Ngôn ngữ lập trình Python phân biệt chữ hoa, chữ thường trong tên. Ví dụ: AB, Ab, aB và ab là các tên khác nhau.

Nhiều ngôn ngữ lập trình, trong đó có Python, phân biệt ba loại tên:

- Tên dành riêng;
- Tên chuẩn;
- Tên do người lập trình đặt.

Tên dành riêng

Một số tên được ngôn ngữ lập trình quy định dùng với *ý nghĩa riêng xác định*, người lập trình không được sử dụng với ý nghĩa khác. Những tên này được gọi là *tên dành riêng* (còn được gọi là *từ khoá*).

Ví dụ: Một số tên dành riêng trong Python: if, else, elif, for, while, def, return, global, in, break,...

Tên chuẩn

Một số tên được ngôn ngữ lập trình dùng với *ý nghĩa nhất định nào đó*. Những tên này được gọi là *tên chuẩn*. Tuy nhiên, người lập trình có thể khai báo và dùng chúng với ý nghĩa và mục đích khác.

Ý nghĩa của các tên chuẩn được quy định trong các *thư viện* của ngôn ngữ lập trình.

Ví dụ: Một số tên chuẩn trong Python: int, float, str, bool, abs, sqrt, pow, round, upper, lower, len, find,…

Tên do người lập trình đặt

Tên do người lập trình đặt được dùng với ý nghĩa riêng, xác định bằng cách khai báo trước khi sử dụng. Các tên này không được trùng với tên dành riêng.

Ví dụ: Tên do người lập trình đặt: a1 delta Tin_hoc

b. Hằng và biến

Hằng

Hằng là các đại lượng có giá trị không thay đổi trong quá trình thực hiện chương trình.

Trong các ngôn ngữ lập trình thường có các hằng số học, hằng lôgic, hằng xâu.

- Hằng số học là các số nguyên hay số thực (dấu phẩy tĩnh hoặc dấu phẩy động), có dấu hoặc không dấu.
- Hằng lôgic là giá trị *đúng* hoặc *sai* tương ứng với *True* hoặc *False*.
- Hằng xâu là chuỗi kí tự trong bảng chữ cái. Khi viết, chuỗi kí tự này được đặt trong cặp dấu nháy đơn ' và ' hoặc cặp dấu nháy kép " và ".

Ví dụ:

- Hằng số học: 2 0 -5 +18
  1.5 -22.36 +3.14159 0.5
  -2.236E03 1.0E-06
- Hằng lôgic:
  + True False
- Hằng xâu:
  + 'Informatic' 'Tin học' 'THPT Xuân Lộc'
  + "Informatic" "Tin học" "THPT Xuân Lộc"

Biến

Biến là đại lượng được đặt tên, dùng để lưu trữ giá trị và giá trị có thể được thay đổi trong quá trình thực hiện chương trình.

Tùy theo cách lưu trữ và xử lí, Python phân biệt nhiều loại biến. Các biến dùng trong chương trình đều phải khai báo. Việc khai báo biến sẽ được trình bày ở các phần sau.

c. Chú thích

Có thể đặt các đoạn chú thích trong chương trình nguồn. Các chú thích này giúp cho người đọc chương trình nhận biết ngữ nghĩa của chương trình đó dễ hơn. Chú thích không ảnh hưởng đến nội dung chương trình nguồn và được chương trình dịch bỏ qua.

Cách ghi chú thích trong Python như sau:

- Ghi chú thích 1 dòng: dùng dấu #

Ví dụ: #Nếu a khác không

- Ghi chú thích trên nhiều dòng: dùng 3 cặp dấu nháy đơn ''' và ''' hoặc 3 cặp dấu nháy đôi """ và """

Ví dụ:

```
'''
  Giải phương trình bậc 1: ax + b = 0
  Nếu a != 0 thì x = -b/a
'''
```

CÂU HỎI VÀ BÀI TẬP

1. Tại sao người ta phải xây dựng các ngôn ngữ lập trình bậc cao?

2. Chương trình dịch là gì? Tại sao cần phải có chương trình dịch?

3. Biên dịch và thông dịch khác nhau như thế nào?

4. Hãy cho biết các điểm khác nhau giữa tên dành riêng và tên chuẩn.

CHƯƠNG II.
CHƯƠNG TRÌNH ĐƠN GIẢN

§3. MỘT SỐ KIỂU DỮ LIỆU CHUẨN, PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN

I. MỘT SỐ KIỂU DỮ LIỆU CHUẨN

Các bài toán trong thực tế thường có dữ liệu vào và kết quả ra thuộc những kiểu dữ liệu quen biết như số nguyên, số thực, kí tự,... Khi lập trình cho những bài toán như vậy, người lập trình sử dụng các kiểu dữ liệu đó thường gặp một số hạn chế nhất định, phụ thuộc vào một số yếu tố như dung lượng bộ nhớ, khả năng xử lí của CPU,...

Vì vậy, mỗi ngôn ngữ lập trình thường cung cấp một số kiểu dữ liệu chuẩn cho biết phạm vi giá trị có thể lưu trữ, dung lượng bộ nhớ cần thiết để lưu trữ và các phép toán tác động lên dữ liệu.

Dưới đây xét một số kiểu dữ liệu chuẩn thường dùng cho các biến đơn trong Python.

1. Kiểu nguyên

int: là một số nguyên, dương hoặc âm, không có số thập phân, có độ dài không giới hạn (chỉ phụ thuộc vào bộ nhớ máy tính).

Ví dụ: -125 0 7

2. Kiểu thực

float: là một số, dương hoặc âm, chứa số thập phân. Giá trị thực lớn nhất mà Python biểu diễn được chỉ phụ thuộc vào bộ nhớ máy tính.

Ví dụ: -2.58 0 789.356 -2.56E03 5.369E-04

3. Kiểu xâu kí tự

str: xâu (chuỗi) các kí tự. Các xâu (chuỗi) trong Python được bao quanh bởi cặp dấu nháy đơn hoặc cặp dấu nháy kép.

Ví dụ: 'Tin học' "Tin học"

4. Kiểu lôgic

bool: nhận một trong hai giá trị True (đúng) hoặc False (sai).

Ghi chú: Người lập trình cần tìm hiểu đặc trưng của các kiểu dữ liệu chuẩn được xác định bởi bộ dịch và sử dụng để khai báo biến.

II. PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN

Để mô tả các thao tác trong thuật toán, mỗi ngôn ngữ lập trình đều xác định và sử dụng một số khái niệm cơ bản: phép toán, biểu thức, gán giá trị cho biến.

Dưới đây sẽ xét các khái niệm đó trong Python.

1. Phép toán

Tương tự trong toán học, trong các ngôn ngữ lập trình đều có những phép toán số học như cộng, trừ, nhân, chia trên các đại lượng thực, các phép toán chia nguyên và lấy phần dư, các phép toán quan hệ,…

Bảng dưới đây là kí hiệu các phép toán đó trong toán và trong Python:

Phép toán	Trong toán học	Trong Python
Các phép toán số học với số nguyên và số thực	+ (cộng), - (trừ), . (nhân), : (chia) div (chia nguyên), mod (lấy phần dư)	+, -, *, /, //, %
Các phép toán quan hệ	< (nhỏ hơn), ≤ (nhỏ hơn hoặc bằng), > (lớn hơn), ≥ (lớn hơn hoặc bằng), = (bằng), ≠ (khác)	<, <=, >, >=, ==, !=
Các phép toán lôgic	¬ (phủ định), ∧ (và), ∨ (hoặc)	not, and, or

Chú ý:

- Kết quả của các phép toán quan hệ cho giá trị lôgic.
- Một trong những ứng dụng của phép toán lôgic là để tạo ra các biểu thức phức tạp từ các quan hệ đơn giản.

2. Biểu thức số học

Trong lập trình, biểu thức số học là một biến kiểu số hoặc một hằng số hoặc các biến kiểu số và các hằng số liên kết với nhau bởi một số hữu hạn phép toán số học, các dấu ngoặc tròn (và) tạo thành một biểu thức có dạng tương tự như cách viết trong toán học với những quy tắc sau:

- Chỉ dùng cặp ngoặc tròn để xác định trình tự thực hiện phép toán trong trường hợp cần thiết;
- Viết lần lượt từ trái qua phải;
- Không được bỏ qua dấu nhân (*) trong tích.

Các phép toán được thực hiện theo thứ tự:

- Thực hiện các phép toán trong ngoặc trước;
- Trong dãy các phép toán không chứa ngoặc thì thực hiện từ trái sang phải, theo thứ tự các phép toán nhân (*), chia (/), chia nguyên (//), lấy phần dư (%) thực hiện trước và các phép toán cộng (+), trừ (-) thực hiện sau.

Ví dụ**:**

Biểu thức trong Toán học	Trong Python
5a+6b	5*a + 6*b

Biểu thức trong Toán học	Trong Python
$\frac{xy}{z}$	x*y/z
$Ax^2 + Bx + C$	A*x*x + B*x + C
$\frac{x+y}{x-\frac{1}{2}} - \frac{x-z}{xy}$	(x + y)/(x - 1/2) - (x - z)/(x*y)

Chú ý:

- Nếu biểu thức chứa một hằng hay biến kiểu thực thì ta có biểu thức số học thực, giá trị của biểu thức cũng thuộc kiểu thực.
- Trong một số trường hợp nên dùng biến trung gian để có thể tránh được việc tính một biểu thức nhiều lần.

3. Hàm số học chuẩn

Để lập trình được dễ dàng, thuận tiện hơn, các ngôn ngữ lập trình đều có thư viện chứa một số chương trình tính giá trị những hàm toán học thường dùng. Các chương trình như vậy được gọi là các *hàm số học chuẩn*. Mỗi hàm chuẩn có tên chuẩn riêng. Đối số của hàm là một hay nhiều biểu thức số học và được đặt trong cặp ngoặc tròn (và) sau tên hàm. Bản thân hàm chuẩn cũng được coi là một biểu thức số học và nó có thể tham gia vào biểu thức số học như một toán hạng (giống như biến và hằng). Kết quả của hàm có thể là nguyên hoặc thực hay phụ thuộc vào kiểu của đối số.

Bảng dưới đây cho biết một số hàm chuẩn thường dùng.

Hàm	Biểu diễn Toán học	Biểu diễn trong Python	Kiểu đối số	Kiểu kết quả
Mũ	x^y	**pow(**x,y**)**	Thực hoặc nguyên	Theo kiểu của đối số
Căn bậc hai	$\sqrt{x}$	**sqrt(**x**)**	Thực hoặc nguyên	Thực
Giá trị tuyệt đối	$\|x\|$	**abs(**x**)**	Thực hoặc nguyên	Theo kiểu của đối số
Lôgarit tự nhiên	lnx	**log(**x**)**	Thực	Thực
Lũy thừa của số e	e^x	**exp(**x**)**	Thực	Thực
Sin	sinx	**asin(**x**)**	Thực	Thực
Cos	cosx	**acos(**x**)**	Thực	Thực

Lưu ý: các hàm nêu trên (ngoại trừ hàm abs) là ở thư viện Toán học, trước khi dùng các hàm này thì phải khai báo là sử dụng thư viện Toán học như sau:

import math

Khi gọi hàm thì phải ghi **math.tên_hàm(<tham số>)**

Chẳng hạn tính $\sqrt{8}$ thì viết như sau: math.sqrt(8)

Ví dụ:

Biểu thức toán học $\frac{-b+\sqrt{b^2-4ac}}{2a}$ trong Python có thể viết dưới dạng:

(-b+math.sqrt(b*b - 4*a*c))/(2*a)

hoặc:

(-b+math.sqrt(math.pow(b,2)- 4*a*c))/2/a

Ngoài những hàm số học chuẩn trên, còn có các hàm chuẩn khác được giới thiệu trong những phần sau.

4. Biểu thức quan hệ

Hai biểu thức cùng kiểu liên kết với nhau bởi phép toán quan hệ cho ta một biểu thức quan hệ.

Biểu thức quan hệ có dạng:

<biểu thức 1> <phép toán quan hệ> <biểu thức 2>

Trong đó, *biểu thức 1* và *biểu thức 2* cùng là xâu hoặc cùng là biểu thức số học.

Ví dụ:

x < 5

i+1 >= 2*j

Biểu thức quan hệ được thực hiện theo trình tự:

- Tính giá trị các biểu thức.
- Thực hiện phép toán quan hệ.

Kết quả của biểu thức quan hệ là giá trị lôgic: *True* (đúng) hoặc *False* (sai).

Trong ví dụ trên, nếu *x* có giá trị 3, thì biểu thức *x* < 5 có giá trị *True*. Nếu *i* có giá trị 2 và *j* có giá trị 3 thì biểu thức *i* + 1 >= 2**j* sẽ cho giá trị *False*.

Ví dụ:

Điều kiện để điểm *M* có toạ độ (*x*; *y*) thuộc hình tròn tâm *I*(*a*; *b*), bán kính *R* là:

math.sqrt((x-a)*(x-a) + (y-b)*(y-b)) <= R

hoặc: math.pow(x-a,2) + math.pow(y-b,2) <= R*R

hoặc: (x-a)**2 + (y-b)**2 <= R**2

5. Biểu thức lôgic

Biểu thức lôgic đơn giản là biến lôgic hoặc hằng lôgic.

Biểu thức lôgic là các biểu thức lôgic đơn giản, các biểu thức quan hệ liên kết với nhau bởi phép toán lôgic. Giá trị biểu thức lôgic là *True* hoặc *False*. Các biểu thức quan hệ thường được đặt trong cặp ngoặc (và).

Phép toán ***not*** được viết trước biểu thức cần phủ định, ví dụ:

not ($x < 1$) thể hiện phát biểu "x không nhỏ hơn 1" và điều này tương đương với biểu thức quan hệ $x >= 1$.

Các phép toán ***and*** và ***or*** dùng để kết hợp nhiều biểu thức lôgic hoặc quan hệ, thành một biểu thức thường được dùng để diễn tả các điều kiện phức tạp.

Ví dụ 1:

Để thể hiện điều kiện $5 \le x \le 11$, trong Python cần phải tách thành phát biểu dưới dạng "$5 \le x$ và $x \le 11$" và được viết như sau:

```
(5 <= x) and (x <= 11)
```

Ví dụ 2:

Giả thiết M và N là hai biến nguyên. Điều kiện xác định M và N đồng thời chia hết cho 3 hay đồng thời không chia hết cho 3 được thể hiện trong Python như sau:

```
((M % 3 == 0) and (N % 3 == 0)) or ((M % 3 !=0) and (N % 3 != 0))
```

6. Câu lệnh gán

Lệnh gán là một trong những lệnh cơ bản nhất của các ngôn ngữ lập trình.

Trong Python câu lệnh gán có dạng:

<tên biến> = <biểu thức>

Trong trường hợp đơn giản, *tên biến* là tên của biến đơn. Kiểu của giá trị biểu thức phải phù hợp với kiểu của biến.

Chức năng của lệnh gán là đặt cho biến có tên ở vế trái dấu "=" giá trị mới bằng giá trị của biểu thức ở vế phải.

Ví dụ:

```
x1 = (-b - math.sqrt(b*b - 4*a*c))/(2*a)
x2 = -b/a - x1
z = z - 1
i = i + 1
```

Trong ví dụ trên, ý nghĩa của lệnh gán thứ ba là giảm giá trị của biến z một đơn vị. Ý nghĩa của lệnh gán thứ tư là tăng giá trị của biến i lên một đơn vị.

Lưu ý: về một số toán tử gán

Toán tử	Ý nghĩa	Ví dụ	Ghi chú
+=	Cộng và gán	N=5 N+=3	N=N+3 ⇨N=8
-=	Trừ và gán	N=5 N-=3	N=N-3 ⇨N=2
*=	Nhân và gán	N=5 N*=3	N=N*3 ⇨N=15
/=	Chia và gán	N=15 N/=3	N=N/3 ⇨N=5
//=	Chia và gán (lấy phần nguyên)	N=19 N//=3	N=N//3 ⇨Kết quả: 6
%=	Chia và gán (lấy phần dư)	N=19 N%=3	N=N%3 ⇨Kết quả: 1
**=	Lấy lũy thừa và gán	N=2 N**=3	N=N**3 ⇨N=8

§4. CẤU TRÚC CHƯƠNG TRÌNH PYTHON

1. Các thành phần của chương trình

Nhắc lại, ngôn ngữ lập trình Python được cấu tạo từ các thành phần cơ bản như: từ khóa, định danh, các câu lệnh, chú thích,…

- **Từ khóa** (tên dành riêng): như đã trình bài ở bài 2, từ khóa là những từ đặc trưng dành riêng trong ngôn ngữ lập trình, chúng ta không thể sử dụng từ khóa để đặt tên biến, tên hàm hoặc bất kỳ định danh nào khác.

- **Định danh**: Định danh là tên được đặt cho các thực thể trong chương trình như lớp, hàm, biến, mảng,… Nó giúp phân biệt thực thể này với thực thể khác trong chương trình. Định danh rất quan trọng bởi nó cho phép trình thông dịch hiểu cần phải truy cập đến đối tượng nào trong chương trình, giống như trong một nhóm thì có tên của các thành viên, nếu các thành viên trùng tên có thể sẽ gây nhầm lẫn và truyền không đúng giá trị làm chương trình bị sai.

- **Câu lệnh**:

+ Các chỉ thị mà trình thông dịch Python có thể thực thi được, được gọi là các câu lệnh. Chẳng hạn x = 1 là một câu lệnh gán có nhiệm vụ đặt giá trị của một biến có tên là x giá trị là 1. Hay các câu lệnh if, for, while,… là các dạng câu lệnh khác để điều khiển chương trình sẽ được trình bày ở các bài sau.

+ Để ngăn cách các câu lệnh, Python sử dụng dấu xuống dòng để phân biệt kết thúc câu lệnh. Do đó, chúng ta không thể tùy tiện sử dụng dấu xuống dòng. Khi muốn ngắt một câu lệnh dài thành nhiều dòng trong chương trình để dễ quan sát, chúng ta phải thêm kí tự đánh dấu tiếp tục câu lệnh \. Chẳng hạn:

```
x = 10 + 11 + 12 + \
    13 + 14 + 15 + \
    16 + 17
```

+ Ngoài ra, trong Python, trình thông dịch sẽ tự hiểu một số trường hợp câu lệnh nhiều dòng, chẳng hạn như câu lệnh nằm trong một dấu ngoặc tròn (), ngoặc vuông [] hoặc ngoặc nhọn {}. Chẳng hạn như sau:

```
x = (10 + 11 + 12 +
    13 + 14 + 15 +
    16 + 17)
```

+ Chúng ta cũng có thể đặt nhiều câu lệnh trên cùng một dòng bằng cách sử dụng dấu chấm phẩy (;), chẳng hạn như sau:

```
n = 78; t = 30; h = 15
```

2. Cấu trúc chương trình

- **Khoảng trắng**: Các kí tự khoảng trắng phổ biến nhất được sử dụng thường là:

Cách ' '

Tab '\t'

Xuống dòng '\n'

- **Khối lệnh và thụt lề**:

Python sử dụng việc thụt lề để đánh dấu (xác định) các khối lệnh. Một khối mã (nội dung của một hàm, vòng lặp,…) được bắt đầu bằng một đoạn thụt đầu dòng và kết thúc bằng dòng ngay kế trên khối thụt lề. Số lượng khoảng trắng thụt lề là tùy thuộc vào người lập trình quy định nhưng nó phải nhất quán trong toàn bộ khối đó. Ví dụ trong cùng một khối lệnh nếu câu lệnh đầu tiên cách vào 1 dấu cách thì câu lệnh thứ hai trong khối cũng phải cách vào một dấu cách, nếu sử dụng hai dấu cách lập tức trình dịch sẽ báo lỗi. *Thông thường, sử dụng 4 dấu cách hoặc 1 tab để thụt lề (4 dấu cách tương đương với 1 tab).*

Ví dụ:

```
s = 0
t = 1
for n in range(1, 11, 1):
    s = s + n
    t = t * n
```

3. Khai báo biến

Cú pháp: **<tên biến> = <giá trị>**

Trong đó:

+ Tên biến: là tên biến mà ta đặt theo quy tắc đặt tên của ngôn ngữ lập trình Python (đã trình bày ở bài 2).

+ Giá trị: là giá trị của biến mà ta muốn gán, kiểu dữ liệu thường là số nguyên, số thực, xâu, lôgic,…

Ví dụ 1: khai báo một biến số nguyên n, gán giá trị là 5.

```
n = 5
```

- Có thể khai báo các biến bằng một giá trị trong một lần khai báo.

Ví dụ 2: khai báo ba biến số nguyên a, b và c, gán giá trị là 10.

```
a = b = c = 10
```

- Có thể khai báo nhiều biến với các giá trị tương ứng của nó trên cùng một dòng.

Ví dụ 3:

```
ten, namsinh, doanvien = 'Nguyễn Văn Hùng', 2004, True
```

Ví dụ 4: Giả sử trong chương trình cần các biến thực a, b, c, d, x1, x2 và các biến nguyên m, n. Khi đó có thể khai báo các biến đó như sau:

```
a = b = c = d = x1 = x2 = 0.0
m = n = 0
```

Một số chú ý khi khai báo biến:

- Cần đặt tên biến sao cho gợi nhớ đến ý nghĩa của biến đó. Điều này rất có lợi cho việc đọc, hiểu và sửa đổi chương trình khi cần thiết.

  Ví dụ, cần đặt tên hai biến biểu diễn điểm toán, điểm tin thì không nên vì ngắn gọn mà đặt tên biến là d1, d2 mà nên đặt là dtoan, dtin hoặc toan, tin.
- Không nên đặt tên biến quá ngắn hay quá dài, dễ mắc lỗi khi viết nhiều lần tên biến. Ví dụ, không nên dùng d1, d2 hay diemmontoan, diemmontin cho điểm toán, điểm tin của học sinh.
- Khi khai báo biến cần đặc biệt lưu ý đến kiểu dữ liệu của nó cho phù hợp với thực tế. Ví dụ, khi khai báo biến để lưu trữ sĩ số học sinh của một lớp thì khai báo kiểu số nguyên nhưng để lưu trữ diện tích của một hình tròn thì khai báo kiểu số thực.

4. Đưa dữ liệu ra màn hình

Cú pháp: **print(<danh sách kết quả ra>)**

Trong đó, *danh sách kết quả ra* có thể là tên biến đơn, biểu thức hoặc hằng. Các hằng xâu thường được dùng để tách các kết quả hoặc đưa ra chú thích. Các thành phần trong kết quả ra được viết cách nhau bởi dấu phẩy (,).

Với hàm print, sau khi đưa các kết quả ra màn hình, con trỏ chuyển xuống đầu dòng tiếp theo. Để con trỏ không chuyển xuống dòng tiếp theo, ta sử dụng thêm tham số **end = ""** hoặc **end = " "** hoặc **end = ''** hoặc **end = ' '**.

Ví dụ:

```
m = 10
n = 15
```

- Xuất giá trị n:

```
print(n)
```

- Xuất giá trị m và n trên cùng một dòng (giá trị m trước, giá trị n sau):

```
print(m, n)              #Python tự động thêm 1 dấu cách giữa 2 số m và n
```

hoặc:

```
print(m, end = ' ')      # Giữa cặp dấu nháy đơn ' và ' có chứa 1 dấu cách
print(n)
```

- Xuất tổng m và n:

```
print('m + n =', m + n)
```

5. Nhập dữ liệu vào từ bàn phím

Cú pháp: **input()**

Giá trị nhập vào của hàm input() thường là kiểu xâu (chuỗi), do đó ta cần chuyển kiểu nếu như muốn lưu trữ giá trị nhập vào không phải kiểu xâu.

Ví dụ 1: Nhập vào họ tên của một học sinh.

```
print('Mời bạn nhập vào họ và tên:')          #Con trỏ xuống đầu dòng tiếp theo
s = input()
```

hoặc:

```
print('Mời bạn nhập vào họ và tên:', end = ' ')       #Con trỏ không xuống dòng
s = input()
```

Ví dụ 2: Nhập vào một số nguyên.

```
print('Nhập một số nguyên:', end = ' ')
n = int(input())     #Chuyển số nguyên nhập vào từ kiểu xâu sang kiểu số
```

Ví dụ 3: Nhập vào hai số thực.

```
print('Nhập hai số thực:')
p = float(input())  #Chuyển số thực nhập vào từ kiểu xâu sang kiểu số
q = float(input())
```

6. Ví dụ chương trình Python đơn giản

Ví dụ 1: Chương trình Python sau đưa các thông báo "Xin chào các bạn!" và "Mời các bạn làm quen với Python." ra màn hình.

```
print("Xin chào các bạn!");
print("Mời các bạn làm quen với Python.");
```

Ví dụ 2: Nhập vào họ tên và năm sinh của một học sinh, sau đó đưa ra màn hình họ tên và tuổi của học sinh vừa nhập.

```
print('Nhập vào họ và tên:', end = ' ')
hoten = input()
print('Nhập vào năm sinh:', end = ' ')
nam = int(input())
print('Họ và tên: ', hoten)
print('Tuổi: ', 2021 - nam)
```

Ví dụ 3: Nhập vào chiều dài và chiều rộng của hình chữ nhật, sau đó tính diện tích và chu vi của hình chữ nhật.

```
print('Nhập chiều dài và chiều rộng của hình chữ nhật:')
d = float(input())
r = float(input())
print('Diện tích của hình chữ nhật là:', d * r)
print('Chu vi của hình chữ nhật là:', (d + r) * 2)
```

§5. SOẠN THẢO, DỊCH, THỰC HIỆN VÀ HIỆU CHỈNH CHƯƠNG TRÌNH

Để có thể thực hiện chương trình được viết bằng một ngôn ngữ lập trình, ta cần soạn thảo, sử dụng chương trình dịch để dịch chương trình đó sang ngôn ngữ máy. Các hệ thống lập trình cụ thể thường cung cấp phần mềm phục vụ cho việc soạn thảo, dịch và hiệu chỉnh chương trình. Dưới đây chỉ giới thiệu cách làm việc với Python:

- Sử dụng hệ điều hành Windows 10 - 32 bit.
- Python phiên bản 3.9.5 - 32 bit.

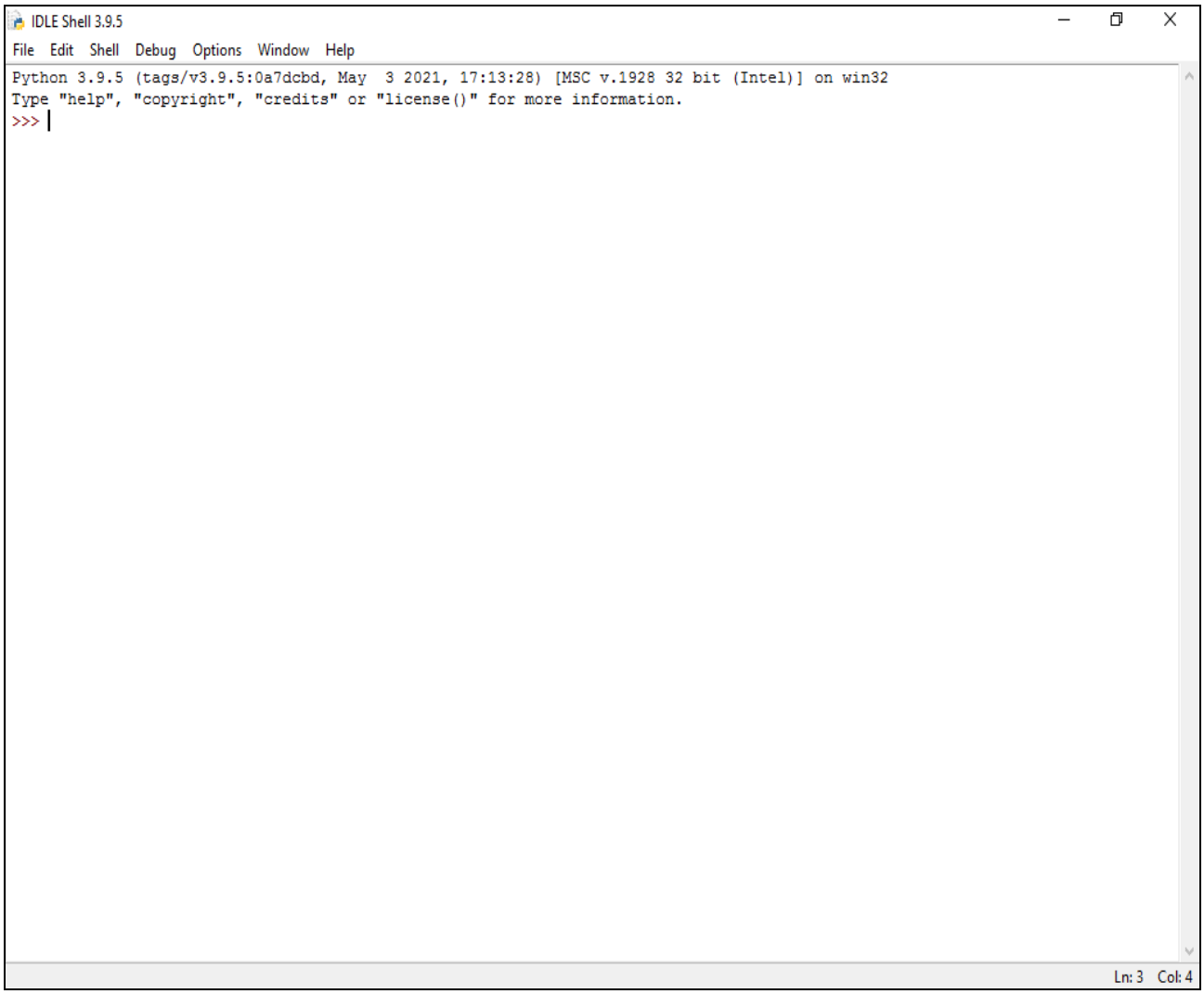
1. Khởi động Python

- Cách 1: Nhắp đúp chuột vào biểu tượng 

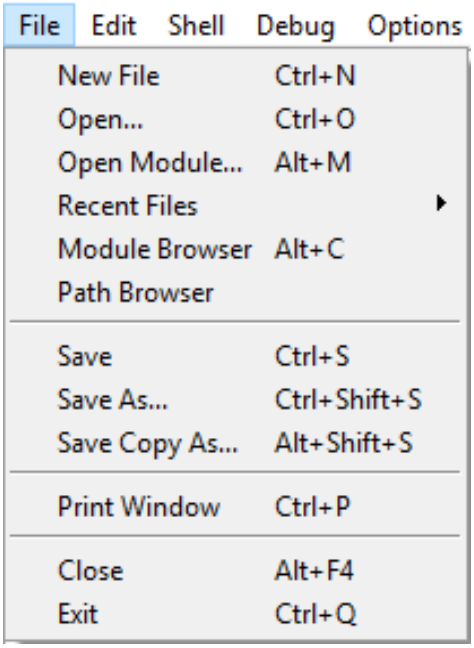
trên màn hình destop.
- Cách 2: Nhắp chuột vào nút Start -> All apps -> Python 3.9 -> IDLE (Python 3.9 32-bit)

2. Làm việc với Python

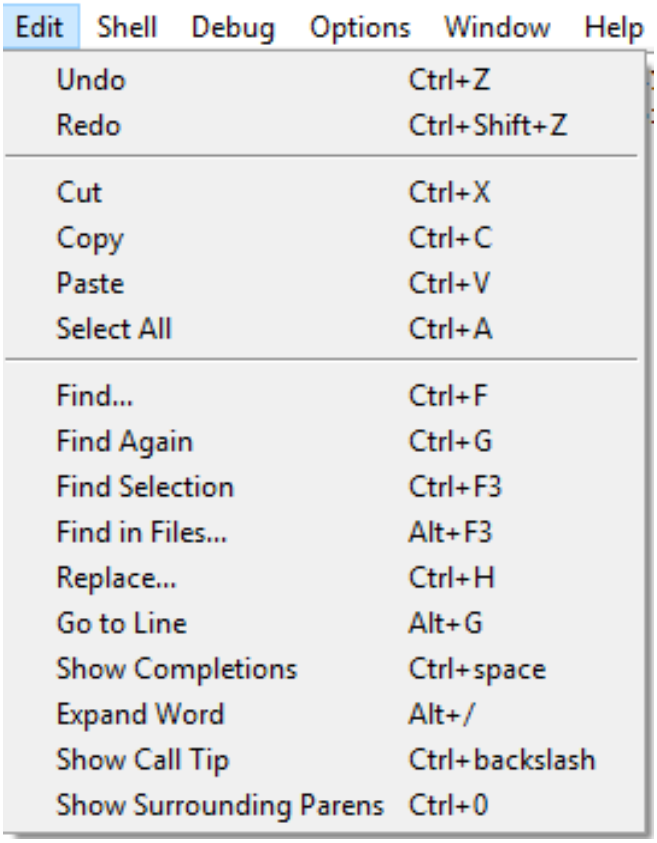
- Giao diện màn hình Python như sau:



- Menu File có các chức năng sau: New File (tạo tệp mới), Open (mở tệp), Save (lưu), Exit (thoát chương trình),...



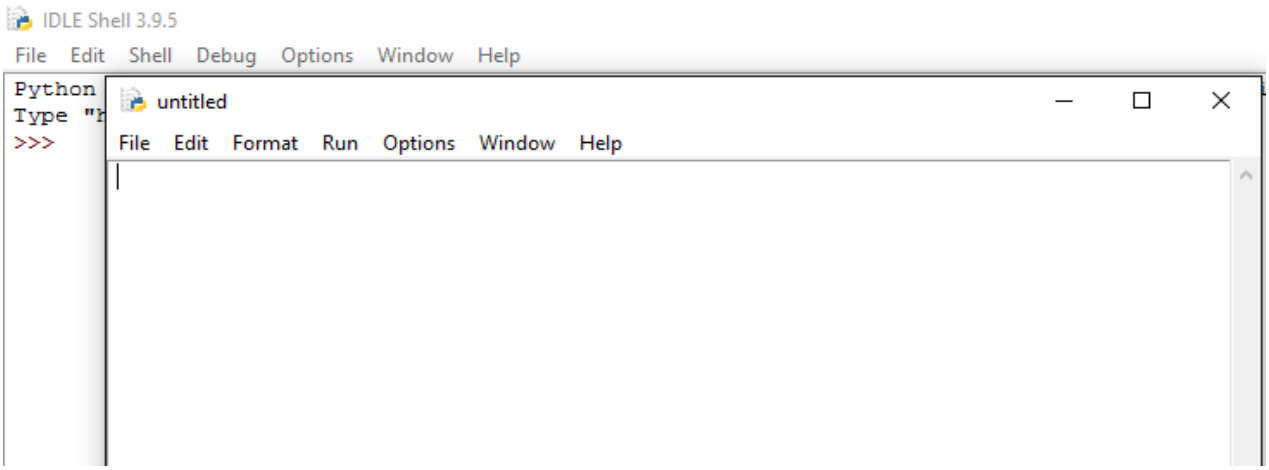
- Menu Edit có các chức năng sau: Cut (cắt), Copy (sao chép), Paste (dán), Find (tìm kiếm), Replace (thay thế),...



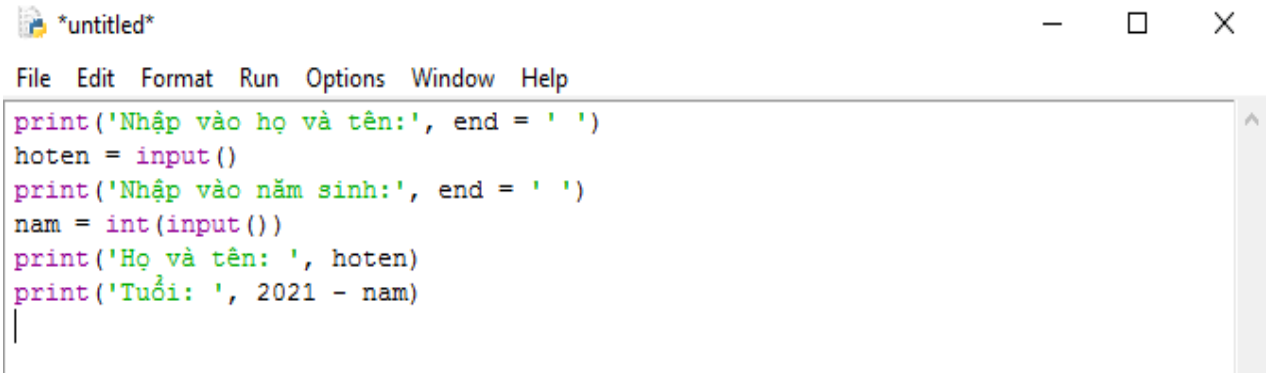
- Tạo tệp mới:

+ File -> New File (hoặc nhấn tổ hợp phím Ctrl+N).

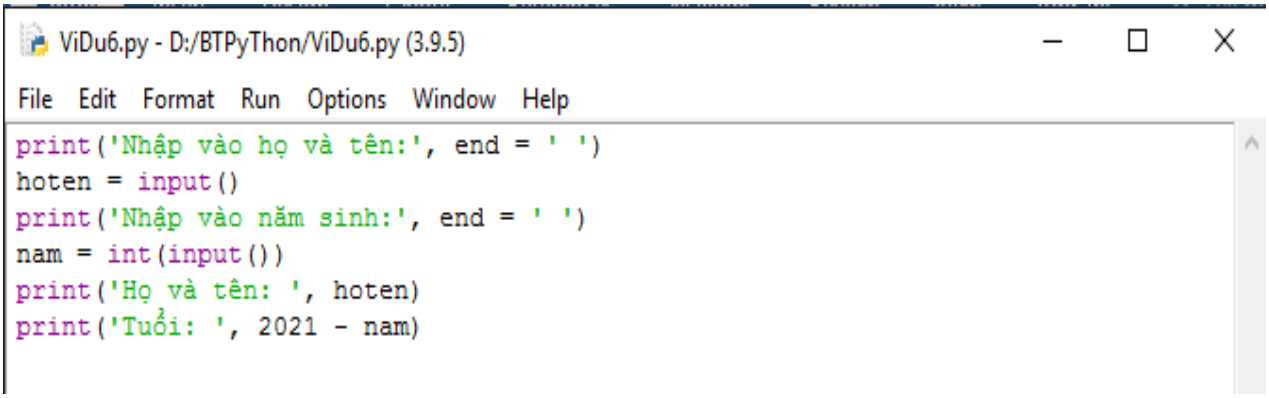
+ Xuất hiện màn hình soạn thảo nội dung như sau:



+ Nhập nội dung các câu lệnh.



+ File -> Save (hoặc nhấn tổ hợp phím Ctrl+S) để lưu: chỉ định ổ đĩa, thư mục, đặt tên tệp (phần mở rộng của tệp là **py**).



+ Chạy chương trình Run -> Run Module (hoặc nhấn phím F5).

BÀI TẬP VÀ THỰC HÀNH 1

1. Mục đích, yêu cầu

- *Giới thiệu một chương trình Python hoàn chỉnh đơn giản;*
- *Làm quen với một số dịch vụ cơ bản của Python trong việc soạn thảo, lưu trữ, dịch và thực hiện chương trình.*

2. Nội dung

a) Gõ chương trình sau:

```
import math        # Sử dụng thư viện Toán học
print('Nhập 3 số thực a, b và c:')
a = float(input())
b = float(input())
c = float(input())
d = b * b - 4 * a * c
x1 = (-b - math.sqrt(d)) / (2 * a)
x2 = -b/a - x1
print('x1 =', x1, ' --- ', 'x2 =', x2)
```

b) Nhấn tổ hợp phím **Ctrl+S** lưu chương trình với tên là **PTB2.py** lên đĩa.

c) Nhấn phím **F5** để thực hiện chương trình. Nhập các giá trị 1; -3 và 2. Quan sát kết quả hiển thị trên màn hình ($x1 = 1.0$; $x2 = 2.0$).

d) Nhấn phím **F5** rồi nhập các giá trị 1; 0; -2.

Quan sát kết quả hiển thị trên màn hình ($x1 = -1.41$; $x2 = 1.41$).

e) Sửa lại chương trình trên sao cho không dùng biến trung gian *d*. Thực hiện chương trình đã sửa với các bộ dữ liệu trên.

f) Sửa lại chương trình nhận được ở mục e bằng cách thay đổi công thức tính $x2$ (có hai cách để tính $x2$).

g) Thực hiện chương trình đã sửa với bộ dữ liệu 1; -5; 6. Quan sát kết quả trên màn hình ($x1 = 2$; $x2 = 3$).

h) Thực hiện chương trình với bộ dữ liệu 1; 1; 1 và quan sát kết quả trên màn hình.

CÂU HỎI VÀ BÀI TẬP

1. Hãy cho biết sự khác nhau giữa hằng có đặt tên và biến.

2. Tại sao phải khai báo biến?

3. Trong Python, nếu một biến chỉ nhận giá trị nguyên trong phạm vi từ 10 đến 25532 thì biến đó có thể được khai báo bằng kiểu dữ liệu nào?

4. Biến p có thể nhận các giá trị 5; 10; 15; 20; 30; 60; 90 và biến x có thể nhận các giá trị 0,1; 0,2; 0,3; 0,4; 0,5. Câu lệnh nhập dữ liệu nào sau đây là đúng?

a)
```
p = float(input())
x = int(input())
```

b)
```
p = bool(input())
x = float(input())
```

c)
```
p = int(input())
x = float(input())
```

d)
```
p = int(input())
x = bool(input())
```

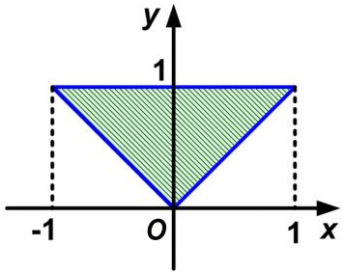
5. Để tính diện tích *S* của hình vuông có cạnh *A* với giá trị nguyên nằm trong phạm vi từ 100 đến 200, cách khai báo *S* nào dưới đây là đúng và tốn ít bộ nhớ nhất?

a) S = True
b) S = 'A * A'
c) S = 0.0
d) S = 0

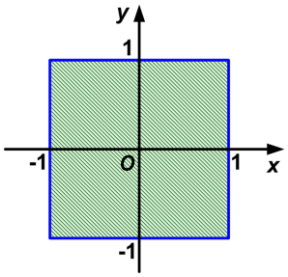
6. Hãy chuyển các biểu thức trong Python dưới đây sang biểu thức toán học tương ứng:

a) a/b*2
b) a*b*c/2
c) 1/a*b/c
d) b/math.sqrt(a*a+b)

7. Hãy viết biểu thức lôgic cho kết quả *True* khi toạ độ (*x*;*y*) là điểm nằm trong vùng gạch chéo kể cả biên của các hình *a* và *b*.



a)



b)

GIỚI THIỆU PYTHON

1. Python là gì?

Guido Van Rossum

Python là một ngôn ngữ lập trình phổ biến. Nó được tạo ra bởi Guido Van Rossum và được phát hành vào năm 1991.

Nó dùng để phát triển web (phía máy chủ), phát triển phần mềm, toán học, kịch bản hệ thống.

2. Python có thể làm gì?

- Python có thể được sử dụng trên máy chủ để tạo các ứng dụng web.

- Python có thể được sử dụng cùng với phần mềm để tạo quy trình công việc.

- Python có thể kết nối với các hệ thống cơ sở dữ liệu. Nó cũng có thể đọc và sửa đổi các tập tin.

- Python có thể được sử dụng để xử lý dữ liệu lớn và thực hiện các phép toán phức tạp.

- Python có thể được sử dụng để tạo mẫu nhanh hoặc phát triển phần mềm sẵn sàng sản xuất.

3. Tại sao lại có tên là Python?

- Python hoạt động trên các nền tảng khác nhau (Windows, Mac, Linux, Raspberry Pi,...).

- Python có một cú pháp đơn giản tương tự như ngôn ngữ tiếng Anh.

- Python có cú pháp cho phép các nhà phát triển viết chương trình với ít dòng hơn một số ngôn ngữ lập trình khác.

- Python chạy trên một hệ thống thông dịch, có nghĩa là mã có thể được thực thi ngay sau khi nó được viết. Điều này có nghĩa là việc tạo mẫu có thể rất nhanh chóng.

- Python có thể được xử lý theo một cách thủ tục, một cách hướng đối tượng hoặc một cách chức năng.

4. Cú pháp Python so với các ngôn ngữ lập trình khác

- Python được thiết kế dễ đọc và có một số điểm tương đồng với ngôn ngữ tiếng Anh với ảnh hưởng từ toán học.

- Python sử dụng các dòng mới để hoàn thành một lệnh, trái ngược với các ngôn ngữ lập trình khác thường sử dụng dấu chấm phẩy hoặc dấu ngoặc đơn.

- Python dựa vào thụt lề, sử dụng khoảng trắng để xác định phạm vi; chẳng hạn như phạm vi của vòng lặp, hàm và lớp. Các ngôn ngữ lập trình khác thường sử dụng dấu ngoặc nhọn cho mục đích này.

HƯỚNG DẪN CÀI ĐẶT PYTHON

B1: Truy cập vào website sau: https://www.python.org/downloads/ hoặc http://thptxuanloc.edu.vn/ (Kéo xuống phần mềm), để tải file cài đặt về máy để cài đặt.

B2. Click đúp vào file mới vừa tải về để cài đặt chương trình.

B3.1 Nếu chọn Install Now khi muốn để ở thự mục mặc định do chương trình đề ra, rồi sau nhấn Close để hoàn thành cài đặt.

B3.2 Nếu chọn Customize installation khi muốn lưu trữ thư mục cài đặt tại nơi khác → Chọn Next → Bấm chọn các tuỳ chọn theo nhu cầu cá nhân, Chọn Browse để chọn đường dẫn đến nơi khác cần lưu chương trình Python → Chọn Install → Nhấn Close để hoàn thành cài đặt.

TÀI LIỆU THAM KHẢO

1. Sách giáo khoa Tin học lớp 11 – tác giả Hồ Sĩ Đàm (chủ biên) – nhà xuất bản giáo dục năm 2007.
2. Website https://www.w3schools.com/python
3. Website https://www.programiz.com/python-programming
4. Website https://www.python.org