CÁC THÀNH PHẦN CƠ BẢN CỦA NGÔN NGỮ PYTHON

Hà Nội, 8-2022

TÍNH TOÁN GIÁ TRỊ BIỂU THỨC ĐƠN GIẢN

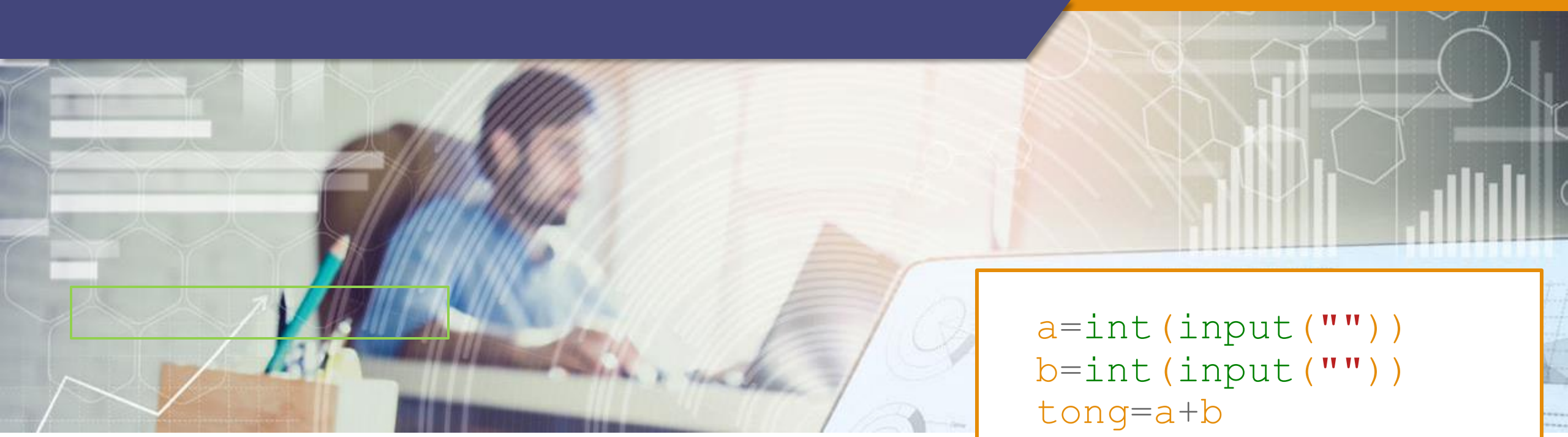
Nhập vào 2 số a và b, tính giá trị các biểu thức:

- Tổng: a+b
- Hiệu: a-b
- Tích: a*b
- Thương: a/b

	Input	Expected	Got	
✔	3 4	7 -1 12 0.75	7 -1 12 0.75	✔
✔	-6 3	-3 -9 -18 -2.0	-3 -9 -18 -2.0	✔
✔	5 2	7 3 10 2.5	7 3 10 2.5	✔
✔	-4 -8	-12 4 32 0.5	-12 4 32 0.5	✔

Chương trình có gì chưa ổn không?

```
a=int(input(""))
b=int(input(""))
tong=a+b
hieu=a-b
tich=a*b
thuong=a/b
print(tong)
print(hieu)
print(tich)
print(thuong)
```



```
2
0
Traceback (most recent call last):
  File "/Volumes/DATA/GIANG DAY/Python cho giao vien
du/vidu2.1.py", line 3, in <module>
    print(a/b)
ZeroDivisionError: division by zero
```

```
1
3
4
-2
3
0.3333333333333333
```

```
a=int(input(""))
b=int(input(""))
tong=a+b
hieu=a-b
tich=a*b
thuong=a/b
print(tong)
print(hieu)
print(tich)
print(thuong)
```

NỘI DUNG

- Từ vựng
- Hằng, biến và biểu thức
- Kiểu dữ liệu
- Câu lệnh nhập xuất dữ liệu
- Xử lý ngoại lệ

HẰNG VÀ BIẾN

TỪ KHÓA TRONG PYTHON

Từ khóa: Là từ những từ được ngôn ngữ lập trình sử dụng cho một mục đích riêng.

- Trong chương trình, không được đặt tên biến trùng với từ khóa

```
False      class      return     is         finally
None       if         for        lambda     continue
True       def        from       while      nonlocal
and        del        global     not        with
as         elif       try        or         yield
assert     else       import     pass
break      except     in         raise
```

HẰNG

- Là các giá trị cố định như các số, kí tự, xâu ký tự.
- Giá trị không thể thay đổi
- Hằng số là các số, viết như thông thường
- Hằng xâu kí tự được đặt trong cặp dấu nháy đơn (‘) hoặc nháy kép (") hoặc nháy ba (‘’’)
- Hằng kiểu logic: True và False

```
>>> print(123)
123
>>> print(98.6)
98.6
>>> print('Hello world’)
Hello world
>>> print(”Hello world")
Hello everyone
>>> print('''Hello world''')
Hello world
```

BIẾN VÀ CÁCH ĐẶT TÊN BIẾN

- **Biến** trong Python là một tên đại diện được dùng để trỏ tới, tham chiếu tới một đối tượng nào đó có trong bộ nhớ.
- Khác định nghĩa biến trong C++/Pascal
- Cách hoạt động tương tự như file và shortcut trong windows.
- Mỗi biến có một kiểu dữ liệu và kích thước, phụ thuộc vào giá trị biến trỏ tới (không khai báo tường minh)

VÍ DỤ. CHƯƠNG TRÌNH SỬ DỤNG BIẾN

Chương trình 1. 1 Ví dụ sử dụng biến

```
1  name = 'Nguyen Van An' # biến kiểu chuỗi
2  age = 40 # biến kiểu số
3  companies = ['FPT', 'Viettel', 'Microsoft'] # biến kiểu
4  danh sách
5  person_tuple = ('Hoang', 'Vienam', 20, 'Student') # biến
   kiểu tuple
6
7  print(name)
8  print(age)
9  print(companies)
10 print(person_tuple)
```

QUY TẮC ĐẶT TÊN BIẾN

- Tên biến được đặt theo quy tắc và gợi nhớ
  - Bắt đầu bằng chữ cái (a..z, A..Z, _), tiếp theo là chữ cái hoặc chữ số (a..z, A..Z, _, 0..9)
  - Không trùng với từ khoá
  - Không được có các kí tự đặc biệt, như: kí trí trống, !, $, #, @,…
  - Có độ dài không giới hạn, tuy nhiên không nên đặt tên quá dài.
  - Phân biệt chữ HOA và chữ thường.
- Ví dụ: count_1, Count_1, first_month, a1, _count là các tên hợp lệ và khác nhau

VÍ DỤ ĐẶT TÊN BIẾN

STT	Tên	Đúng/Sai
1	Birthday	
2	Too_hot?	
3	First_Initial	
4	1stprogram	
5	down.to.earth	
6	see you	
7	OldName	
8	raise	
9	One+Two	

VÍ DỤ ĐẶT TÊN BIẾN

STT	Tên	Đúng/Sai
1	Birthday	Đ.
2	Too_hot?	S. (Không được có dấu “?”)
3	First_Initial	Đ.
4	1stprogram	S. (Không được bắt đầu bằng số)
5	down.to.earth	S. (Không được có dấu “.”)
6	see you	S. (Không được có dấu cách)
7	OldName	Đ. (Tốt, nên viết hoa đầu từ tiếng Anh)
8	raise	S. (Trùng từ khoá)
9	One+Two	S. (Không được có dấu “+”)

KINH NGHIỆM ĐẶT TÊN BIẾN

Một số cách đặt tên biến:

- Đặt tên biến gợi nhớ
- Nếu có nhiều từ thì nối với nhau bằng kí tự gạch dưới: my_school_name, my_flower,…
- Viết hoa các chữ cái đầu tiên của một từ: MyFuntion, YourSchoolName,…
- Viết thường từ đầu tiên và viết hoa chữ cái đầu của các từ tiếp theo, ví dụ: myName, yourSchoolName,…

QUY ƯỚC ĐẶT TÊN BIẾN TRONG PYTHON

- Tên lớp thường bắt đầu bằng chữ hoa; các định danh khác bắt đầu với chữ thường.
- Biến private bắt đầu với một dấu dạch dưới ‘_’
- Biến có mức độ private cao hơn bắt đầu với 2 dấu gạch dưới __
- Biến có tên bắt đầu và kết thúc bằng 2 dấu gạch dưới (Ví dụ: __init__) là tên đặc biệt được ngôn ngữ định nghĩa.

BIẾN VÀ PHÉP GÁN

- Biến và phép gán là những thành phần cơ bản và thường gặp nhất trong bất kỳ ngôn ngữ lập trình nào.
- Biến trong Python bắt buộc phải tạo ra cùng phép gán
- Kiểu dữ liệu của biến được xác định dựa trên giá trị được gán cho biến.
- Python là một ngôn ngữ định kiểu động và định kiểu yếu.

BIẾN VÀ PHÉP GÁN

Biến = <Biểu thức>

- <Biểu thức>: có thể là hằng, là biến, là một biểu thức phức hợp
- Tính toán giá trị của <biểu thức> và gán giá trị đó cho biến
- Vế trái có thể có nhiều hơn 1 biến, hai biến cách nhau bởi dấu phẩy
- Biến được tạo ra khi thực hiện phép gán

```
1  >>> a = 1000
2  >>> b = a
3  >>> c = b
4  >>>
```

BIỂU THỨC

- Biểu thức là sự kết hợp của một hay nhiều giá trị, hằng, biến, toán tử và lời gọi hàm
- Ngôn ngữ lập trình diễn giải và tính toán biểu thức để tạo ra một giá trị mới.
- Giá trị trả về của biểu thức thuộc các kiểu dữ liệu tương ứng

BIẾN VÀ PHÉP GÁN

```
>>> a = b = c = 1000
>>> a
1000
>>> b
1000
>>> c
1000
>>> fname, age = "Huu Hoang", 40
>>> fname
'Huu Hoang'
>>> age
40
>>>
```

```
>>> a = 1000 # ban đầu a trỏ tới giá trị số
>>> a
1000
>>> a = "Huu Hoang" # a trỏ tới giá trị chuỗi
>>> a
'Huu Hoang'
```

```
#khai báo và gán giá trị cho biến a, b, c
a = b = c = 1000
#gán giá trị của biểu thức cho biến x
x = 2 * (a + b) + c
# gán giá trị cho nhiều biến
fname, age = "Jonh Smith", 40
#đưa giá trị của biến ra màn hình
print(a)
print(x)
print(fname," ",age)
```

KIỂU DỮ LIỆU

KIỂU DỮ LIỆU

- Trong Python, các biến, hằng đều có “kiểu”, nhưng không cần phải khai báo kiểu của biến khi sử dụng
- Kiểu số trong python chia làm 3 loại: số nguyên (int), số thực (float), số phức (complex).
  - Số nguyên (int): -14, -2, 0, 1, 100, 401233
  - Số thực dấu phẩy động (float):

    -2.5 , 0.0, 98.6, 14.0
  - Số phức (complex): 3 + 2j, 5 + 3J
- Kiểu logic (bool) bao gồm hai giá trị **True** hoặc **False** và là một kiểu con của kiểu số nguyên

KIỂU DỮ LIỆU

- Kiểu xâu ký tự được viết trong cặp dấu nháy đơn ‘, nháy kép “, hoặc ba dấu nháy đơn hoặc ba dấu nháy kép.
- Python tự động phân biệt số nguyên, số thực với một xâu kí tự
- Ví dụ
- “+” có nghĩa là ”phép cộng” nếu toán hạng là số
- “+” có nghĩa là “phép nối” nếu toán hạng là một xâu kí tự

```
1  s1 = 'Hello world' # t
2  s2 = "Hello world" # t
3  s3 = '''Hello world'''
4  s4 = """Hello world"""
```

```
>>> a = 1 + 4
>>> a
5
>>> a = "Hello" + "world"
>>> a
'Helloworld’
>>>
```

KIỂU DỮ LIỆU

- Python tự động nhận biết kiểu của biến, hằng và biểu thức trong chương trình
- Không thể thực hiện phép toán đối với biến có kiểu dữ liệu khác nhau. Ví dụ: "cộng 1" vào một xâu kí tự
- Hàm type() để kiểm tra kiểu dữ liệu của biến

```
>>> eee = 'hello ' + 'there'
>>> eee = eee + 1
Traceback (most recent call
last):  File "<stdin>", line 1,
in <module>TypeError: Can't
convert 'int' object to str
implicitly
>>> type(eee)
<class'str'>
>>> type('hello')
<class'str'>
>>> type(1)
<class'int'>
>>>
```

CÁC PHÉP TOÁN SỐ HỌC

Toán tử	Mô tả	Ví dụ
+	Phép cộng hai toán hạng	x + y = 5
-	Phép trừ hai toán hạng	x – y = -1
*	Phép nhân hai toán hạng	x * y = 8
/	Phép chia hai toán hạng.	x / y = 1.9
%	Phép chia lấy dư	17 % 2 = 1
**	Phép toán mũ (x ** y tương đương x^y)	2 ** 3 = 8
//	Phép chia lấy thương nguyên	9 // 2 = 4

CÁC PHÉP TOÁN SO SÁNH

Toán tử	Mô tả	Ví dụ
==	So sánh bằng. Nếu hai toán hạng bằng nhau sẽ trả về giá trị True.	3 == 5
!=	So sánh khác. Trả về True nếu hai toán hạng có giá trị khác nhau.	x != 2
>	Lớn hơn. Trả về True toán hạng bên trái lớn hơn toán hạng bên phải.	x+y > z
>=	Lớn hơn hoặc bằng. Trả về True toán hạng bên trái lớn hơn hoặc bằng toán hạng bên phải.	x*y >= 0
<	Nhỏ hơn. Trả về True toán hạng bên trái nhỏ hơn toán hạng bên phải.	x < y+z
<=	Nhỏ hơn hoặc bằng. Trả về True toán hạng bên trái nhỏ hơn hoặc bằng toán hạng bên phải.	z <= 5

CÁC PHÉP TOÁN LOGIC

Toán tử	Mô tả	Ví dụ
and	Phép toán AND, cho kết quả True khi cả hai toán hạng đều có giá trị True	p and q cho kết quả False
or	Phép toán OR, cho kết quả True khi một trong hai toán hạng có giá trị True	p or q cho kết quả True
not	Phép phủ định, cho kết quả ngược lại với giá trị của toán hạng	not p cho kết quả False

CÁC PHÉP TOÁN THAO TÁC TRÊN BIT

Toán tử	Mô tả	Ví dụ
&	Phép toán AND bit	p & q = 12 ($12 = 00001100_2$)
|	Phép toán OR bit	p|q = 61 ($61 = 00111101_2$)
^	Phép toán XOR bit	p^q = 49 ($49 = 00110001_2$)
~	Phép đảo ngược bit	(~p) = -61
<<	Phép toán dịch trái chuỗi bit	p << 2 = 240 ($240 = 1111000_2$)
>>	Phép toán dịch phải chuỗi bit	p >> 2 = 15 (15 = 00001111_2)

VÍ DỤ VỀ PHÉP TOÁN

Hãy cho biết kết quả của các câu lệnh sau đây?

```
print(3*2 + 10/2)
```

```
print("doremi"*2 + "doremon"*3)
```

```
x = 1 + 2 * 3 - 4 / 5 ** 6
```

PHÉP TOÁN VÀ THỨ TỰ ƯU TIÊN

Toán tử	Mô tả
()	Mở đóng ngoặc, các phép toán trong mở đóng ngoặc có mức ưu tiên cao nhất.
**	Phép toán lấy mũ
+x, -x, ~x	Phép toán cộng, trừ một ngôi; phép nghịch đảo bit
*, /, //, %	Nhân, chia, chia lấy nguyên, chia lấy dư
+, -	Cộng, trừ
<<, >>	Phép dịch bít sang trái, sang phải
&	Phép toán AND bit
^	Phép toán XOR bit
|	Phép toán OR bit
==, !=, >, >=, <=	Các phép toán so sánh
not	Phép toán logic NOT
and	Phép toán logic AND
or	Phép toán logic OR

PHÉP TOÁN VÀ THỨ TỰ ƯU TIÊN

Một số lưu ý khi sử dụng biểu thức và phép toán:

- Ghi nhớ quy tắc từ trên xuống dưới
- Sử dụng dấu ngoặc khi viết code
- Sử dụng biểu thức toán học đơn giản và dễ hiểu nhất có thể
- Chia nhỏ các biểu thức

BÀI TẬP THỰC HÀNH

Bài 2-1. Viết chương trình thực hiện đổi số giây cho trước sang số ngày, số giờ, số phút, số giây và in kết quả ra màn hình.

Ví dụ: nếu số giây là 684500 thì kết quả in ra màn hình như sau:

684500 giây = 7 ngày 22 giờ 8 phút 20 giây

CÂU LỆNH NHẬP XUẤT DỮ LIỆU

XUẤT DỮ LIỆU

print(<string>)

Trong đó:

- <string> là một hoặc nhiều chuỗi kí tự, hai chuỗi kí tự cách nhau bởi dấu phẩy;
- <string> có thể bao gồm hằng kí tự, giá trị của một biến, một biểu thức.
- Để định dạng được chuỗi giá trị cần đưa ra cho lệnh print, chúng ta có thể sử dụng nhiều kiểu định dạng khác nhau %-formatting, str.format() hoặc f-strings

XUẤT DỮ LIỆU – ĐỊNH DẠNG f-string

- Dễ dùng => được ưa chuộng trong cộng đồng Python
- Sử dụng trong câu lệnh *print()* với một kí tự 'f' đứng trước xâu cần in
- Các biểu thức nằm trong cặp ngoặc nhọn '{' và '}' sẽ được tính toán và chuyển thành xâu và thay thế vào vị trí biểu thức đó

```
>>> hoten = "Nguyen Thanh Van"

>>> tuoi = 40

>>> print(f"Thong tin cua can bo la:
\n Ho ten: {hoten} \nTuoi: {tuoi}")

Thong tin cua can bo la:

 Ho ten: Nguyen Thanh Van
 Tuoi: 40
```

XUẤT DỮ LIỆU – ĐỊNH DẠNG f-string

Hiển thị dấu phẩy động

- Giá trị dấu phẩy động có hậu tố f
- Chỉ độ chính xác: số chữ số thập phân bằng một giá trị nằm ngay sau ký tự dấu chấm
- Giá trị thập phân sẽ được tự động làm tròn

```
>>> x = 1.234567
>>> print(f"{x:.2f}")
1.23
>>> x = 1.2567889
 >>>print(f"{x:.2f}")
1.26
```

XUẤT DỮ LIỆU – ĐỊNH DẠNG f-string

Xác định độ rộng

- f-string có thể tự động điền khoảng trống hoặc kí tự khác vào nếu số lượng kí tự ít hơn độ rộng được chỉ định

```
>>> ten = "Thanh Huyen"
>>> tuoi = 5
>>> print(f"{ten:20} {tuoi:5}")
Thanh Huyen                5
>>> print(f"ten} {tuoi}")
Thanh Huyen 5
```

XUẤT DỮ LIỆU – ĐỊNH DẠNG f-string

Căn lề

- f-string có thể kết hợp xác định độ rộng với căn lề dữ liệu
- Các dấu được sử dụng:
  - <: căn lề trái
  - >: căn lề phải
  - ^: căn giữa

```
s1 = 'a'

s2 = 'ab'

s3 = 'abc'

s4 = 'abcd'

print(f'{s1:>10}')

print(f'{s2:>10}')

print(f'{s3:>10}')

print(f'{s4:>10}')
```

```
         a
        ab
       abc
      abcd
```

BÀI TẬP THỰC HÀNH

Bài 2-2. Viết chương trình thực hiện công việc sau:

- Cho phép nhập vào tên, hệ số lương và tiền lương của 2 nhân viên
- Hiển thị thông tin của nhân viên theo định dạng như ví dụ dưới đây

```
Ten: Nguyen Thi Thanh Huyen
He so luong: 3.999999
Tien luong: 1999999.9999
Ten: Le ANh
He so luong: 3.66
Tien luong: 100000000.9
Danh sach nhan vien
-------------------------------------------------------------
Nguyen Thi Thanh Huyen                4.00          2000000.00
Le ANh                                3.66        100000000.90
```

NHẬP DỮ LIỆU

<Ten_bien> = input([<thong_bao>])

- <thong_bao> là một một chuỗi kí tự sẽ được hiển thị trên màn hình cho người nhập dữ liệu.
- Người dùng kết thúc nhập bằng phím Enter, toàn bộ dữ liệu vừa nhập sẽ được coi như một xâu kí tự và trả về cho biến

```
1  hoten = input("Cho biet ho ten: ")
2  namsinh = input("Cho biet ngay thang nam sinh: ")
```

NHẬP DỮ LIỆU

- Sử dụng hàm chuyển kiểu để chuyển sang kiểu dữ liệu mong muốn
- Lỗi sẽ xảy ra nếu xâu chứa kí tự không phải kí tự số

```
1  a = int(input("Hay nhap vao mot so nguyen: "))
2  b = float(input("Hay nhap vao mot so thuc: "))
```

```
>>> sval = '123'
>>> type(sval)
<class 'str'>
>>> print(sval + 1)
Traceback (most recent call
last):  File "<stdin>", line 1,
in <module>
TypeError: Can't convert 'int'
object to str implicitly
>>> ival = int(sval)
>>> type(ival)
<class 'int'>
>>> print(ival + 1)
124
```

BÀI TẬP THỰC HÀNH

Bài 1-1. Thực hiện các công việc sau đây

- Tạo file mã nguồn có tên bai1-1.py chứa mã nguồn chương trình thực hiện việc hiển thị các dấu “*” theo dạng dưới đây.
- Chạy chương trình và xem kết quả

SỬ DỤNG MODULE

SỬ DỤNG MODULE

- Sử dụng lại code
  - Các hàm viết sẵn (Routines) có thể được gọi nhiều lần trong một chương trình
  - Các hàm viết sẵn có thể dùng cho nhiều chương trình
- Phân chia bởi không gian tên (namespace)
  - Nhóm dữ liệu với hàm để sử dụng dữ liệu dễ dàng hơn
- Cài đặt dịch vụ và dữ liệu dùng chung
  - Cung cấp cấu trúc dữ liệu dùng chung cho các chương trình con (subprogram)

SỬ DỤNG MODULE

- Module là các hàm và biến được định nghĩa trong file riêng
- Được đưa vào chương trình bằng các từ khóa from hoặc import

```
from module import function
function()
```

```
import module
module.function()
```

- Module chính là không gian tên namespace
  - Có thể sử dụng để tổ chức tên biến. Ví dụ:
    - `atom.position = atom.position - molecule.position`

SỬ DỤNG MODULE

- 3 cách truy cập tới module bằng lệnh import:

```
import math
print math.sqrt(2.0)
```

hoặc:

```
from math import sqrt
print sqrt(2.0)
```

hoặc :

```
from math import *
print sqrt(2.0)
```

- Có thể import nhiều module trên một dòng:

```
import sys, string, math
```

- Mỗi dòng chỉ được dùng một lệnh "**from x import y**"

MỘT SỐ MODULE THÔNG DỤNG

- Một số module thường dùng:
  - **math**: các hàm toán học
  - **random**: các hàm ngẫu nhiên
  - **collections:** các kiểu dữ liệu container trong python
  - **os**: các hàm liên quan đến hệ điều hành
  - **string:** các hàm liên quan đến xâu kí tự
  - **datetime**: Các hàm ngày giờ

BÀI TẬP THỰC HÀNH

Bài 2-4. Viết chương trình nhập vào một số nguyên và kiểm tra số đó có phải số chính phương hay không? Gợi ý: Sử dụng hàm sqrt của thư viện math.