

第一周平时作业

朱士杭 231300027

2024 年 2 月 29 日

1 问题一

1.1 强类型语言与弱类型语言

1.1.1 强类型语言

对于一种语言而言如果里面的变量被定义为某一种数据类型之后，除非人为对其进行强制类型转换，否则其数据类型不会发生改变，那么该语言就称为强类型语言

1.1.2 弱类型语言

如果这种语言里面的变量被定义之后，在不同的场景之下可以灵活类型转换，不会在一开始特别定义其数据类型

1.2 动态类型语言与静态类型语言

1.2.1 动态类型语言

程序在运行期间才会检查数据类型，在写脚本的时候不需要对变量进行类型声明，比如说 python 是解释器不需要编译那么在遇到一个变量定义的时候比如说 `a=3.0` 自动将 `a` 的数据类型定义为 `float`

1.2.2 静态类型语言

程序在编译期间就会检查所有的数据类型，在写脚本的时候需要对每个变量进行显示的类型声明，比如说 C++ 中 `double a=3.0` 需要对 `a` 的数据类型进行说明以便于编译器之后进行类型的检查

ps 补充说明:问题一开始并不知道答案,参考过相关资料,链接如下:<https://zhuanlan.zhihu.com/p/6257035> 但是答案都是自己看完以后自己总结之后写上去的养成学术诚信的好习惯

2 问题二

2.1 编译执行和解释执行

2.1.1 编译执行

编译器会先将脚本转换为编译语言，然后生成经过绑定联结之后生成一个 exe 的可执行文件，此时就会脱离原来的脚本，即使电脑没有配置相应的语言也可以执行

优点是编译执行效率高，占用资源小，适合复杂程序的开发

缺点严重依赖环境与操作系统，比如在 Windows 上的编译程序直接迁移到 Linux 系统上会执行不了

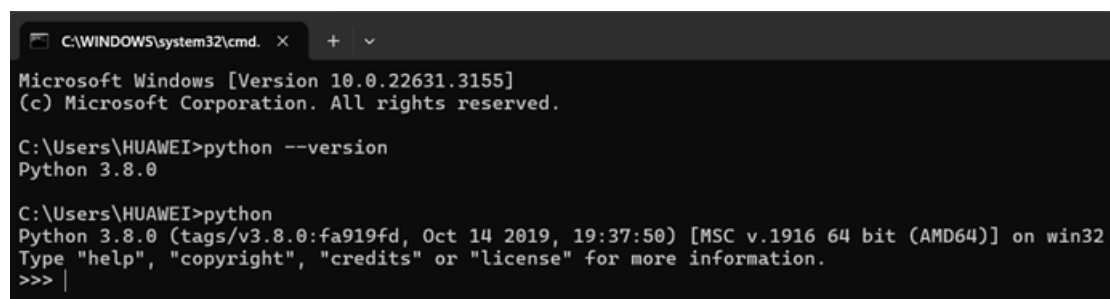
2.1.2 解释执行

不需要编译器，程序会根据脚本一行一行执行下去，不会生成相应的可执行文件，如果出现 bug 会立即终止程序，需要有相应的语言环境比如 python

优点是执行的时候不依赖于环境

缺点是一句一句执行会造成计算机资源的浪费，执行效率低

3 问题三



```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HUAWEI>python --version
Python 3.8.0

C:\Users\HUAWEI>python
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

图 1: python3.8 版本查看

ps 补充说明：自己在电脑里面安装了四个版本的 python，分别是 3.8、3.9、3.11、3.12 但是不知道为什么 cmd 里面只显示 3.8 的版本（估计是默认版本吧），之后又切换到了 3.11 的文件夹下，显示出 3.11 版本（用的是 pycharm 和 anaconda 配置的环境）

```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HUAWEI>E:

E:\>cd PythonANACONDA

E:\PythonANACONDA>python --version
Python 3.11.5

E:\PythonANACONDA>python
Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

图 2: python3.11 版本查看

4 问题四——不同方式使用 Python

```
1 num1=int(input("请输入第一个数字: "))
2 num2=int(input("请输入第二个数字: "))
3 sum=num1+num2
4 print("{}与{}相加结果为: {}".format(*args: num1,num2,sum))
```

运行 第一次作业 x

E:\PythonANACONDA\python.exe E:\大学课程\大一下\人工智能程序设计\

请输入第一个数字: 3

请输入第二个数字: 4

3与4相加结果为: 7

进程已结束, 退出代码为 0

图 3: 用文件方式编写的相加

```
Python 控制台 x
E:\PythonANACONDA\python.exe "E:/PyCharm Community Edition 2023.2.3/p
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['E:\大学课程\大一下\人工智能程序设计\作业集'])

>>> Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:2
In [3]: num2=4
In [4]: num1+num2
Out[4]: 7
In [5]: |
```

```
10 num1 = (int) 3
10 num2 = (int) 4
> 特殊变量
```

图 4: 用交互模式编写相加

5 问题五——使用 help 命令

```
E:\Python\ANACONDA>python
Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help(input)
Help on built-in function input in module builtins:

input(prompt=' ', /)
    Read a string from standard input. The trailing newline is stripped.

    The prompt string, if given, is printed to standard output without a
    trailing newline before reading input.

    If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return), raise EOFError.
    On *nix systems, readline is used if available.
```

图 5: help(input)

```
>>> help(int)
Help on class int in module builtins:

class int(object)
|   int([x]) -> integer
|   int(x, base=10) -> integer
|
|   Convert a number or string to an integer, or return 0 if no arguments
|   are given. If x is a number, return x.__int__(). For floating point
|   numbers, this truncates towards zero.
|
|   If x is not a number or if base is given, then x must be a string,
|   bytes, or bytearray instance representing an integer literal in the
|   given base. The literal can be preceded by '+' or '-' and be surrounded
|   by whitespace. The base defaults to 10. Valid bases are 0 and 2-36.
|   Base 0 means to interpret the base from the string as an integer literal.
|   >>> int('0b100', base=0)
|   4
|
|   Built-in subclasses:
|       bool
|
|   Methods defined here:
|
```

图 6: help(int)

```
>>> help(str)
Help on class str in module builtins:

class str(object)
|   str(object='') -> str
|   str(bytes_or_buffer[, encoding[, errors]]) -> str
|
|   Create a new string object from the given object. If encoding or
|   errors is specified, then the object must expose a data buffer
|   that will be decoded using the given encoding and error handler.
|   Otherwise, returns the result of object.__str__() (if defined)
|   or repr(object).
|   encoding defaults to sys.getdefaultencoding().
|   errors defaults to 'strict'.
|
|   Methods defined here:
|
|   __add__(self, value, /)
|       Return self+value.
|
|   __contains__(self, key, /)
|       Return key in self.
```

图 7: help(str)

6 后记

这是我第一次使用 LaTeX 写作业，不足之处敬请谅解，之后会不断提升使用 LaTeX 的技术，在此纪念一下第一次使用这样的工具，就像我们第一次用 Python 和 C++ 打印出 “HelloWorld” 一样弥足珍贵