

Bài tập 01

Họ và tên: Thái Nguyên Quốc

MSSV: 079206015807

Câu 1: Tại sao mình học lập trình

Mình học lập trình với nhiều mục đích cụ thể:

- **Tò mò và muốn hiểu cách máy tính hoạt động:** Từ nhỏ mình luôn thắc mắc các ứng dụng và trò chơi vận hành ra sao, lập trình giúp mình “nhìn bên trong” những thứ ấy.
- **Tự tay tạo ra sản phẩm:** Việc biến ý tưởng thành một ứng dụng, một trang web hay một công cụ thực tế mang lại cảm giác rất thú vị và tự hào.
- **Rèn luyện tư duy logic và giải quyết vấn đề:** Lập trình giống như luyện não mỗi ngày; mình học cách phân tích, chia nhỏ vấn đề và tìm hướng giải nhanh, gọn, chính xác.
- **Cơ hội nghề nghiệp và thu nhập:** Công nghệ thông tin phát triển mạnh, kỹ năng lập trình giúp mở rộng cánh cửa việc làm tốt và ổn định cho tương lai.
- **Khả năng học hỏi không giới hạn:** Công nghệ thay đổi liên tục, nên lập trình cũng cho mình động lực học suốt đời, không ngừng cập nhật cái mới.

Câu 2: Cách máy tính thực thi lệnh

Khi một chương trình chạy, bên trong CPU diễn ra một chu trình xử lý lệnh liên tục, vừa chặt chẽ vừa cực nhanh.

1. **Lấy lệnh (Fetch):** Bộ đếm chương trình (Program Counter) chỉ tới địa chỉ lệnh kế tiếp trong bộ nhớ. CPU gửi yêu cầu, bộ nhớ trả về lệnh, và lệnh được đưa vào thanh ghi lệnh.

2. **Giải mã (Decode):** Khởi điều khiển đọc mã lệnh, xác định thao tác cần làm và các dữ liệu liên quan. Ở bước này, CPU như đang “hiểu ý” của lập trình viên.
3. **Thực thi (Execute):** Nếu lệnh là tính toán, đơn vị số học–logic (ALU) sẽ xử lý; nếu là truy xuất dữ liệu, CPU sẽ đọc hoặc ghi từ bộ nhớ; nếu là lệnh rẽ nhánh, bộ đếm chương trình được cập nhật.
4. **Ghi kết quả (Write Back):** Kết quả được lưu vào thanh ghi hoặc trả về bộ nhớ chính.

Bốn bước trên diễn ra liên tục hàng tỷ lần mỗi giây. Các CPU hiện đại còn dùng **pipeline** (chồng chéo nhiều lệnh), **bộ nhớ đệm cache** và **thực thi ngoài thứ tự** để tăng tốc. Nhờ vậy, mọi hành động của chúng ta, từ bấm phím, mở ứng dụng, cho tới xử lý video, đều được hoàn thành trơn tru gần như tức thì.

Câu 3: Trình biên dịch và trình thông dịch

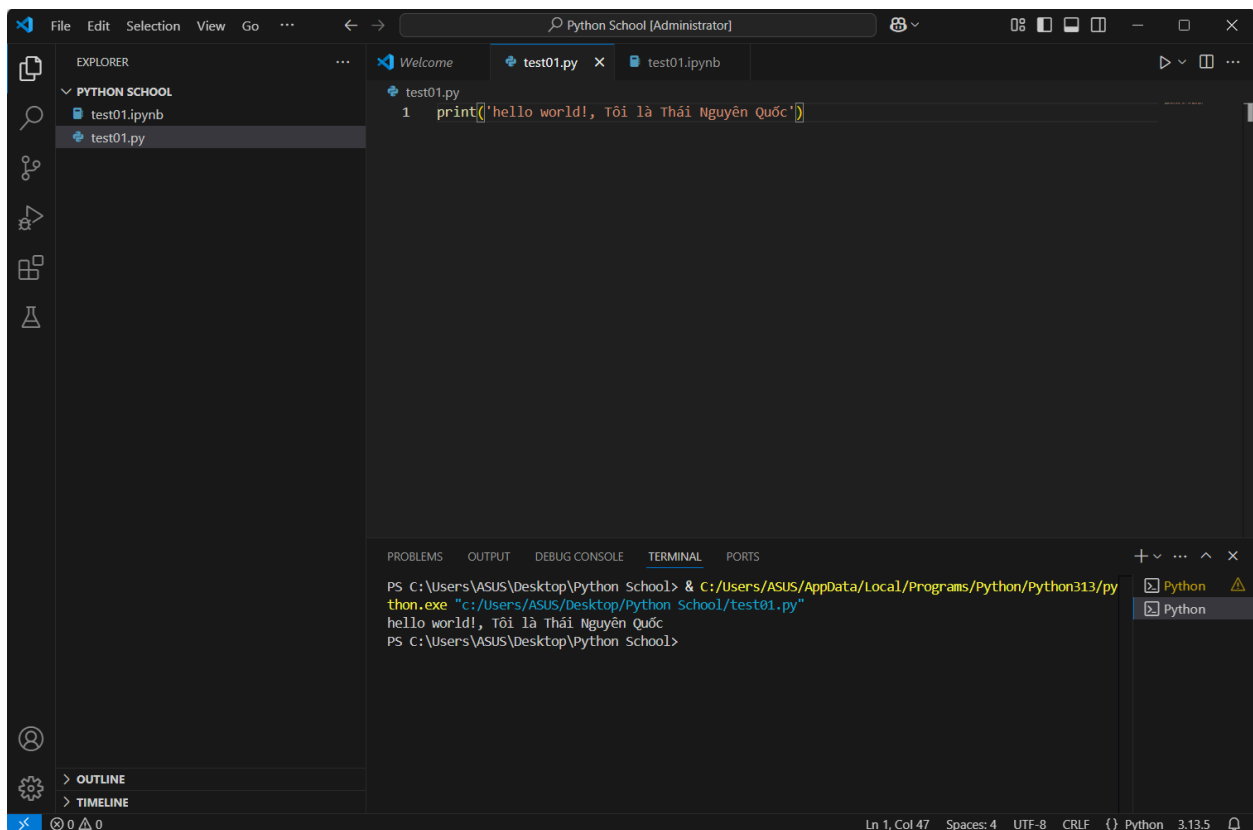
- **Trình biên dịch (Compiler):** Dịch toàn bộ mã nguồn sang mã máy một lần, tạo ra tệp thực thi độc lập. Chương trình chạy nhanh vì đã được chuyển hoàn toàn sang ngôn ngữ máy. Tuy nhiên, khi sửa mã, ta phải biên dịch lại.
- **Trình thông dịch (Interpreter):** Đọc và thực thi từng dòng lệnh ngay lập tức, không cần tạo tệp thực thi. Điều này tiện cho việc thử nghiệm, nhưng tốc độ chạy chậm hơn.

Ngoài ra, nhiều ngôn ngữ hiện đại kết hợp cả hai: **biên dịch thành mã trung gian** rồi thông dịch hoặc biên JIT (Just-In-Time) khi chạy, ví dụ Java và C#.

Bảng phân loại ngôn ngữ lập trình

Loại ngôn ngữ	Đặc điểm chính	Ví dụ
Biên dịch	Dịch toàn bộ trước khi chạy, tốc độ cao	C, C++, Rust, Go, Pascal
Thông dịch	Thực thi từng dòng, dễ thử nghiệm, chạy chậm hơn	Python, JavaScript, Ruby, PHP
Kết hợp (biên + thông dịch/JIT)	Biên sang bytecode, sau đó thông dịch hoặc JIT	Java (JVM), C# (.NET)

1. Câu 4: Viết chương trình in ra màn hình dòng chữ: “Hello world! Tôi là xxx”. Trong đó xxx là tên sinh viên



The screenshot shows the Visual Studio Code interface with a Python file named `test01.py` open. The code in the file is:

```
1 print('hello world!, Tôi là Thái Nguyên Quốc')
```

The terminal at the bottom shows the command to run the script and its output:

```
PS C:\Users\ASUS\Desktop\Python School> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/Python School/test01.py"
hello world!, Tôi là Thái Nguyên Quốc
PS C:\Users\ASUS\Desktop\Python School>
```

The status bar at the bottom indicates the file is at line 1, column 47, using UTF-8 encoding with CRLF line endings, and the Python version is 3.13.5.