# Tips

## Loại câu hỏi

Ở mức độ nâng cao (level middle trở lên), thì NTD (nhà tuyển dụng) sẽ đánh giá ứng viên (UV) qua những câu hỏi WHY, WHEN, HOW nhiều hơn là WHAT. VD, thay vì hỏi "WHAT are smart pointers in C++, their types and benefits?" thì NTD sẽ hỏi "HOW a smart pointer works? HOW it knows WHEN to release an object memory? WHY auto\_ptr is bad? Ngoài ra những câu hỏi SO SÁNH (concepts/technique … tương đồng nhau) cũng được hỏi nhiều.

## Cẩn thận với những gì ghi trong CV

Nếu tự đánh giá mình *advanced* về algorithms, thì NTD có thể hỏi sâu về algorithms, hoặc thậm chí code thử cho họ xem." Do đó, cái gì không chắc thì nên tự đánh giá ở mức *basic* (hoặc bỏ khỏi CV).

## Ghi nhớ những technique và skills đã dùng trong các dự án cũ

NTD hay thắc mắc những dự án cũ mà UV đã làm. Nếu chỉ mô tả những task đã thực hiện trong dự án (vốn đã ghi sẵn trong CV) thì chưa đủ. NTD sẽ hỏi sâu hơn, đại loại: chức năng tiêu biểu, chức năng thấy khó nhất/thú vị nhất, dùng technique gì cho chức năng đó … Nếu UV nói trúng điểm họ thích thú, họ có thể hỏi kỹ hơn nữa.

Do đó, nên **tự REVIEW KỸ lại những dự án cũ** trước khi interview. Nếu không trả lời tốt, họ có thể sẽ đánh giá UK kém vì không nắm rõ được technical của những dự án đã làm.

## Nói chuyện với tốc độ vừa phải

Điều này vừa giúp câu giờ, vừa giúp diễn đạt suy nghĩ trong đầu một cách trôi chảy và súc tích hơn (đặc biệt quan trọng khi nói tiếng Anh). Đừng để xảy ra tình huống UV nắm chắc kiến thức, nhưng vì hấp tấp trong lời nói mà diễn đạt không trọn vẹn hoặc dùng từ không chuẩn.

## Đừng trả lời ngay sau khi nhận được câu hỏi

Hãy dành ra tầm 2-3s suy nghĩ về câu hỏi trước để chắc chắn hiểu đúng câu hỏi, sau đó mới trả lời. Nếu chưa hiểu rõ câu hỏi, nhờ NTD nhắc lại theo cách này hoặc cách khác; hoặc chủ động trình bày cách hiểu của bản thân để nhờ NTD xác nhận.

## Tỏ ra học hỏi

Đối với những câu hỏi mà không trả lời được, thì nói với họ là: "Đó là câu hỏi hay. Mình sẽ note lại câu hỏi và tìm câu trả lời sau".

## Tỏ ra hài hước

Dù có bị bắt bẻ đến bế tắc, thì cũng phải chèn vào vài câu đùa như: "Câu hỏi này khó ". Và cườiiii. Việc nhỏ này sẽ giúp UV thả lỏng hơn.

## Xin hint

Nếu UV trả lời không tốt hoặc không hiểu rõ ngụ ý hỏi của NTD, thì đừng ngần ngại xin gợi ý. VD: Có thể hỏi lại theo cách khác? Xin ví dụ để hiểu hơn về câu hỏi … Ngoài ra, đôi khi câu hỏi của NTD khá chung chung, thì có thể nhờ họ breakout câu hỏi.

## Nhắc đến các technique nâng cao mà UV hiểu rõ

Trong câu trả lời, UV có thể liên hệ đến những concept nâng cao (tất nhiên phải đảm bảo hiểu rõ về nó) để show NTD. Đây cũng là cách câu giờ để hướng NTD đến những kiến thức mà mình biết, hơn là hàng ngàn câu hỏi khác của NTD.

## Tóm lại, phải chuẩn bị THẬT KỸ trước mỗi lần interview

NTD có cả nghìn câu hỏi. Họ luôn ở trong thế chủ động, còn UV thì bị động. Họ luôn tìm cách bắt bẻ để dành được lợi thế khi deal lương. Cho dù UV có nhiều kinh nghiệm làm dự án, mà không chuẩn bị kỹ, thì câu trả lời cũng khó mà trôi chảy, mạch lạc, chính xác được. Chưa kể, làm nghề càng lâu, CV càng bao gồm nhiều skill và phạm vi kiến thức càng rộng, thì càng dễ "nhớ cái này, quên cái kia".

Ngoài ra, UV đừng nghĩ rằng khi mình có nhiều năm kinh nghiệm rồi, thì NTD sẽ không hỏi kỹ về technical, mà chỉ tập trung về project, management skills hay soft skills. Điều này sai hoàn toàn. Đúng là ngôn ngữ chỉ là phương tiện, nhưng vấn đề là NTD đâu có biết UV là ai, trình độ như thế nào, skill ghi trong CV có trung thật hay không, technical có phù hợp với dự án của họ không, tư duy có tốt không, … Nên tất cả UV đều sẽ được hỏi nhiều về technical, chẳng qua là mức độ khó của câu hỏi sẽ khác nhau tùy vào trình độ và level ứng tuyển.

Tất nhiên cũng có NTD hỏi rất ít về mặt ngôn ngữ, mà tập trung về technical và softskills. Nên UV cũng k thể đoán được. Nên tóm lại, là **cái nào cũng phải ôn**.

Thời gian để thật sự chuẩn bị kỹ sẽ **không dưới 4 tuần** (giả định 4h/ngày).

## Đừng bỏ quên thuật toán

Nhiều NTD đánh giá UV qua tư duy logic nhiều hơn là ngôn ngữ hay dự án. Nên bắt buộc UV phải ôn kĩ thuật toán. Nhưng chỉ cần ôn những thuật toán CƠ BẢN NHẤT thôi. Bởi vì buổi PV thường chỉ 1-2 tiếng. Nên phần thuật toán chỉ chiếm tầm 15 phút, vừa đủ để code xong một bài thuật toán cơ bản.

## Dò la HR về kiến thức phỏng vấn

Đối với các job mà công ty nhờ đơn vị HR (bên thứ 3) tuyển dụng, thì UV nên chủ động hỏi thử HR về các kiến thức trong cuộc PV. Vì có khả năng cao trước đó có người PV rồi và HR đã hỏi họ về câu hỏi rồi. HR nào có kinh nghiệm thì họ sẽ chủ động nói cho UV luôn. Hai bên đều có lợi mà!

Links tham khảo:

[[Chuẩn Bị Phỏng vấn] Đi làm lập trình ở nước ngoài - YouTube](https://www.youtube.com/watch?v=-xql5Y35-SM&ab_channel=D%E1%BA%A1yNhauH%E1%BB%8Dc)

# Technical Interview

## General

* Why did you leave your latest company?
* You said that you have some problems with your latest company. Can you tell me a bit more about these problems?
* If you pass the exam, how long it takes for you to join our company?
* Why do you want to join our company?
* In addition to questions that we already asked, do you have any great skills that you have not shown yet?

## CV

* Can you tell me some hightlights about your previous projects?
* What did you learn the most from each of your previous projects?
* In your CV, you described yourself to be "advanced" in **design patterns**. So can you list some design patterns you’re used?
* Can you tell me more about the Singleton design pattern? What is it? Why it is helpful? **How to implement** it? How to apply the pattern in multithread program? …
* In your CV, you described yourself to be "advanced" in **algorithms and data structures**. So can you tell steps to find a square root of any number?
* Do you have experiences when working with **multiprocess and multithread**? Can you tell me possible issues which are usually caused by multithreading?
* Can you specify the difference between **mutex and samaphore**? When to use mutex and when to use samaphore?
* In all of your previous projects, **which projects do you like** (or learn) most? In this project, which is the feature that you find most interesting (or most difficult).

## C++

* Can you compare the differences between a Stack and a Queue. Can you implement a Stack and some of its basic operations in C++? I sent you a link to an online text editor; now please share your screen.
* Do you have experences when working with smart pointers in C++? Can you tell me how does a smart pointer work?

## Code Review

You said that you usually review your code’s members. Can you review this piece of code snippet below?

{

var x = new Bonus(5);

var y = new Refund(1);

var z = new Transfer(3);

var l = new List {x, y, z};

Foreach (var a in l)

{

If (a is Bonus)

Write("user is bonus");

If (a is Refund)

Write("user is refund");

If (a is Transfer)

Write("user transfer");

}

}

**Reviewed Points:**

* Should not create the list "l" with keyword "new". Reason: "x", "y", "z" are already allocated in memory.
* Should use bracket { } in the "if" statements.
* Should replace "if … if … if" with "if … else if … else if …".
* Should add a default case for checking value of "a" and put it in an "else".
* Should apply Polymorphism instead of using conditional statements. This helps the code open for extensions but closed for modification.

Fixed version:

// Bonus, Refund, Transfer, Withdraw are base classes of "BankOperation"

{

var x = new Bonus(5);

var y = new Refund(1);

var z = new Transfer(3);

var k = new Withdraw(2);

var l = List {x, y, z, k};

Foreach (var a in l)

{

a.Operation();

}

}

# Director Interview

* What do you expect when working with us?
* Why did you leave your latest company?
* What is your salary expectation?
* Are you willing to negotiate if we cannot offer your salary as you expected?
* What is your expected career path in the future?
* You said that you wanna be a PM in the future. Do you have any plan to achieve that? Or what can we do to help you to achieve your goal?
* What is your strongest points?
* What is your weakest points?
* After all of these years, what is the most important thing that you think you learnt?

# Should-Asked Questions

* How long can I know the final **interview result**?
* How about the **current team**? (team size, level …)
* When the new project **will be started**?