

Username: Pralay Patoria **Book:** Coding Interviews: Questions, Analysis & Solutions. No part of any chapter or book may be reproduced or transmitted in any form by any means without the prior written permission for reprints and excerpts from the publisher of the book or chapter. Redistribution or other use that violates the fair use privilege under U.S. copyright laws (see 17 USC107) or that otherwise violates these Terms of Service is strictly prohibited. Violators will be prosecuted to the full extent of U.S. Federal and Massachusetts laws.

Integer Value from a String

Interviewer: You mentioned in your résumé that you are proficient on C/C++. How many years have you used these two languages?

Candidate: It has been six or seven years since I learned them at my university.

Interviewer: Cool, it sounds like you are a veteran. Let's discuss some C++ problems. (The interviewer hands a piece of paper with the source code from [Listing 9-1](#) to the candidate.) What is the output when this piece of code executes?

Listing 9-1. C++ Code for the Member Initialization Order

```
class A {

private:

    int n1;

    int n2;

public:

    A(): n2(0), n1(n2 + 2) {

    }

    void Print() {

        std::cout << "n1: " << n1 << ", n2: " << n2 << std::endl;

    }

};

int main(int argc, char* argv[]) {

    A a;

    a.Print();


    return 0;

}
```

Candidate: (Reads code for a while) `n1` is 2, and `n2` is 0.

Interviewer: Why?

Candidate: In the initialization list of the constructor, `n2` is initialized as 0, so the result of `n2` is 0. And then `n1` is initialized with `n2+2`, so it is 2.

 **Note:** This answer is NOT correct. Please refer to the comments at the end of this section for more details.

Interviewer: Are members initialized according to their order in the initialization list?

Candidate: (Confused) I am not quite sure.

Interviewer: No problem. Let's move on to another problem. What is the usage for the function `atoi` in the C library?

Candidate: It converts a numeric string to an integer. For example, if the input string is "123", it returns 123.

Interviewer: That's right. Please write a function `StrToInt` to convert a string to an integer. Of course, it is not allowed to call `atoi` or other similar functions in the library. Are there any questions?

Candidate: (Smiles with confidence) No problem.

The candidate writes down the code in [Listing 9-2](#) on paper in a short period of time.

Listing 9-2. C++ Code to Convert a String to an Integer (Version 1)

```
int StrToInt(char* string) {

    int number = 0;

    while(*string != 0) {

        number = number * 10 + *string - '0';

        ++string;

    }

    return number;

}
```

Candidate: I have finished.

Interviewer: Oh, that was quick. (Reads the code quickly) Do you think it is complete? Scrutinize your code.

Candidate: (Reads the code from the beginning) Sorry that I forgot to verify the `NULL` pointer for the input string.

The candidate adds two lines of code. The revised code is shown in [Listing 9-3](#).

Listing 9-3. C++ Code to Convert a String to an Integer (Version 2)

```
int StrToInt(char* string) {
    if(string == NULL)
        return 0;

    int number = 0;

    while(*string != 0) {
        number = number * 10 + *string - '0';
        ++string;
    }

    return number;
}
```

Interviewer: Have you finished? (Reads the code again) Your output is 0 when the input string is `NULL`. What is the output when the input is a string "0"?

Candidate: It is also 0.

Interviewer: It returns 0 for two different cases. How do you distinguish these cases when its caller gets a 0?

Candidate: (Confused) I do not know.

Interviewer: Similar to the library function `atoi`, we may distinguish them with a global variable. When the input is invalid, the variable is set to a special value. However, it is not set when the input is "0". When the caller gets a 0 from the function `StrToInt`, it knows what happens according to the global variable.

Candidate: Oh, I see. (Picks up the pen) Let me rewrite that.

Interviewer: Wait. Are there any other invalid inputs besides the `NULL` pointer?

Candidate: (Thinking, with sweat on forehead) If a string has some characters beyond the range from '0' to '9', it is invalid.

Interviewer: Are all characters beyond the range from '0' to '9' invalid?

Candidate: The positive sign ('+') and negative sign ('-') should be valid.

Interviewer: Correct. Think carefully before you start to write code.

The candidate thinks for a few minutes, and writes down the code shown in [Listing 9-4](#).

Listing 9-4. C++ Code to Convert a String to an Integer (Version 3)

```
enum Status {kValid= 0, kInvalid};

int g_nStatus = kValid;

int StrToInt(const char* str) {
    g_nStatus = kInvalid;

    int num = 0;

    if(str != NULL) {
        const char* digit = str;
        bool minus = false;

        if(*digit == '+')
            digit++;
        else if(*digit == '-') {
            digit++;
            minus = true;
        }

        while(*digit != '\0') {
            if(*digit >= '0' && *digit <= '9') {
                num = num * 10 + (*digit - '0');
                digit++;
            }
            else {
                num = 0;
                break;
            }
        }
    }
}
```

```

        if(*digit == '\0') {

            g_nStatus = kValid;

            if(minus)

                num = 0 - num;

        }

    }

    return num;

}

```

Interviewer: Can you explain your code?

Candidate: I defined a global variable `g_Status` to mark whether the input is valid. It is initialized to the invalid status and is set to valid only when all characters in the string are converted smoothly. It is possible for a positive or negative sign to appear in the first character, so the first character of the input string is handled specially. Any non-digital character after a positive or negative sign indicates an invalid input string, so the conversion stops immediately.

Interviewer: It sounds good. Is there anything missing?

Candidate: (Thinks for about two minutes) Is it necessary to handle overflow issues?

Interviewer: Isn't it necessary? It seems that there is no more time left for you to make changes again. I am going to leave the last minutes for you to ask a few questions. Do you have any questions to me?

Candidate: What is the salary package at your company?

Interviewer: What is your expectation?

Candidate: Many of my classmates got a package of more than a hundred thousand dollars per year. I do not want to have less than them.

Interviewer: The package of the entry level for graduates is determined by our human resources department, so I cannot answer your questions, but I will forward your expectation to the recruiter.

Candidate: That is OK. Thank you.

Interviewer: Any more questions?

Candidate: No.

Interviewer: Cool. That is the end of this round of interview. Thank you.

The Interviewer's Comments

I was very disappointed when the candidate provided an incorrect answer to the question about the initialization list in a C++ constructor because he mentioned that he was proficient on C++. The initialization list is a commonly used concept in C++. Data members in an initialization list are initialized in the order of the member variable declarations in the class. Since `n1` was declared before `n2` in the given code, `n1` should be initialized before `n2`. When `n1` was initialized with `n2+2`, `n2` had not been initialized yet, and it might be a random value. Therefore, the value of `n1` was random after initialization, and then `n2` was initialized as 0.

The next question was a coding interview problem to convert a string to an integer. It looked like a simple problem, and the candidate implemented it within less than 10 lines of code at first. However, what I expected was complete and robust code, which should handle cases including a normal numeric string, a `NULL` pointer, an empty string, a string with non-digital characters, a string with a positive or negative sign, as well as overflow issues. It was not necessary for the candidate to implement a function identical to `atoi`, but he should define the behavior of his function with various inputs and explain it to me explicitly. The most serious problem for the candidate was that he did not have a habit of considering all possible inputs in advance before writing code, so his code was problematic and incomplete. After I gave him many hints, there still were many bugs left in his code.

[Listing 9-5](#) contains the sample code, which covers more cases than the candidate's code:

Listing 9-5. C++ Code to Convert a String to an Integer (Version 4)

```

enum Status {kValid= 0, kInvalid};

int g_nStatus = kValid;

int StrToInt(const char* str) {

    g_nStatus = kInvalid;

    long long num = 0;

    if(str != NULL && *str != '\0') {

        bool minus = false;

        if(*str == '+')

            str ++;

        else if(*str == '-') {

            str ++;

            minus = true;

        }

        if(*str != '\0') {

            num = StrToIntCore(str, minus);

```

```

    }

}

return (int)num;

}

long long StrToIntCore(const char* digit, bool minus) {

    long long num = 0;

    while(*digit != '\0') {

        if(*digit >= '0' && *digit <= '9') {

            int flag = minus ? -1 : 1;

            num = num * 10 + flag * (*digit - '0');

            if((!minus && num > 0x7FFFFFFF)

                || (minus && num < (signed int)0x80000000)) {

                num = 0;

                break;

            }

            digit++;

        }

        else {

            num = 0;

            break;

        }

    }

    if(*digit == '\0') {

        g_nStatus = kValid;

    }

    return num;

}

```

What the candidate cared about was salary during the Q & A phase. Often employees who only care about salary are prone to be job-hoppers. Additionally, his expectation was based on his classmates' salary packages. Did it indicate that he was lacking self-evaluation skills?

To sumup, my opinion was not to hire the candidate because he currently does not have the competence to write complete and robust code.

Source Code:

106_StringToInt.cpp

Test Cases:

- Normal Test Cases (Numeric strings for positive/negative numbers or zero; strings with non-digital characters besides '+' and '-')
- Boundary Test Cases (Strings for the maximal and minimal integers)
- Robustness Test Cases (A `NULL` point to a string; an empty string)