# Number Systems and Coding Representations

## 1. Conversion of Decimal Numbers

Using your birthdate (29-07-2004), the first three digits are 290. Below are the conversions of 290 to binary, octal, and hexadecimal representations:

### Binary:

To convert 290 into binary, divide the number by 2 and record the remainders until the quotient becomes 0. Write the remainders in reverse order.  
Steps:  
- 290 ÷ 2 = 145, remainder = 0  
- 145 ÷ 2 = 72, remainder = 1  
- 72 ÷ 2 = 36, remainder = 0  
- 36 ÷ 2 = 18, remainder = 0  
- 18 ÷ 2 = 9, remainder = 0  
- 9 ÷ 2 = 4, remainder = 1  
- 4 ÷ 2 = 2, remainder = 0  
- 2 ÷ 2 = 1, remainder = 0  
- 1 ÷ 2 = 0, remainder = 1  
Binary: 100100010

### Octal:

To convert 290 into octal, divide the number by 8 and record the remainders until the quotient becomes 0. Write the remainders in reverse order.  
Steps:  
- 290 ÷ 8 = 36, remainder = 2  
- 36 ÷ 8 = 4, remainder = 4  
- 4 ÷ 8 = 0, remainder = 4  
Octal: 442

### Hexadecimal:

To convert 290 into hexadecimal, divide the number by 16 and record the remainders until the quotient becomes 0. Write the remainders in reverse order. For remainders above 9, use hexadecimal letters.  
Steps:  
- 290 ÷ 16 = 18, remainder = 2  
- 18 ÷ 16 = 1, remainder = 2  
- 1 ÷ 16 = 0, remainder = 1  
Hexadecimal: 122

## 2. Importance of Proficiency in Number Systems

Proficiency in number systems is critical in technical jobs because:  
- Hardware and software interaction: Computers operate in binary, while humans often work in decimal. Bridging this gap is essential.  
- Data encoding and communication: Different devices and systems might require octal, hexadecimal, or binary formats.  
- Troubleshooting: Debugging errors often involves interpreting memory addresses or error codes, which are commonly represented in hexadecimal.

## 3. Importance of Different Coding Representations

Using representations like ASCII, Unicode, or BCD is necessary because:  
- Data consistency: Different devices and systems may support only specific encoding standards.  
- Compatibility: Unicode supports multiple languages, while ASCII is limited to English characters.  
- Efficiency: Some coding systems are optimized for specific data types (e.g., BCD for numerical data).

### Choosing a Representation:

- ASCII: Ideal for basic text in English.  
- Unicode: Best for multilingual projects.  
- BCD: Useful for numerical data in financial systems.

### Example:

In an international e-commerce website, product descriptions must support multiple languages. Using Unicode ensures compatibility with all languages, while ASCII would fail for non-English text.

## 4. Comparison of Coding Representations

|  |  |  |  |
| --- | --- | --- | --- |
| Representation | Character Set Size | Compatibility | Usage |
| ASCII | 128 characters | Limited to English | Text data in simple systems |
| Unicode | 1.1M+ characters | Universal | Multilingual text |
| Gray Code | Binary sequence | Specialized | Error correction in circuits |
| BCD | Decimal digits (0–9) | Limited | Financial and numerical data |
| EBCDIC | 256 characters | IBM mainframes | Legacy systems |

## 5. Recommendation

For this task, Unicode is the best choice because it offers broad compatibility and supports multilingual data, making it suitable for modern applications that require global reach.