# Lesson 25 ASCII

Everything a computer “sees” is merely a **series of numbers**, including English letters (a, b, c, etc.) and number system symbols (decimal, octal, hexadecimal, and binary).

Any **character** holds a **value** (an integer number) rather than that character itself.

For example, number **0** corresponds to another number (**48**).

```
int x;
```

For the computer, **uppercase** and **lowercase** letters are considered different because they correspond to different values. So, if we used **X** in the code, an error would appear in the program.

In this lesson, we’ll discuss **ASCII**, which **holds values** for alphabetic letters, numbers, and special additional characters.

```
int main() {
        int s = 'A';
        printf(" %d ", s);
}
```

**output:**

```
65
```

If we only type the character, it functions as a **variable**.

Instead, if we type it enclosed in **single quotes ‘ ‘**, then it is a **character** with a certain value for the computer.

Computers can only store numbers ( no letters ). So, when a computer prints a letter, the computer recognizes it by knowing the **ASCII value** that represents it.

Here, we assigned **s** the ASCII value for the letter **A**

We can print this value by doing the following:

```
int main() {
        printf(" %d ", 'A');
}
```

**output :**

```
65
```

We enclosed the letter in **single quotes** to print the **ASCII value** that represents it.

### A program to print the uppercase alphabets and their ASCII values:

```
int main() {
        int i;
        for (i = 'A'; i <= 'Z'; i++)
                printf(" %c %d ", i, i);
}
```

**output :**

```
A 65 B 66 C 67 D 68 E 69 F 70 G 71 H 72 I 73 J 74 K 75 L 76 M
77 N 78 O 79 P 80 Q 81 R 82 S 83 T 84 U 85 V 86 W 87 X 88 Y
89 Z 90
```

Try to code yourself:

- - > click here: lesson 25 ASCII - Replit

### A program to print the lowercase alphabets and their ASCII values:

```
int main() {
        int i;
        for (i = 'a'; i <= 'z'; i++)
                printf(" %c %d ", i, i);
}
```

**output:**

```
a 97 b 98 c 99 d 100 e 101 f 102 g 103 h 104 i 105 j 106 k 107 l
108 m 109 n 110 o 111 p 112 q 113 r 114 s 115 t 116 u 117 v 118 w 119
x 120 y 121 z 122
```

**A program to print the ASCII values for numbers range from 0 to 9:**

```
int main() {

int i;

for (i = '0'; i <= '9'; i++)

printf(" %c\t%d \n", i, i);

}
```

**output:**

```
0 48

1 49

2 50

3 51

4 52

5 53

6 54

7 55

8 56

9 57
```

Try to code yourself:

- - > click here: lesson 25 ASCII - Replit

We put **0** and **9** between **single quotes**: **'0'** , **'9'**

**48** represents number **0**

**47** represents number **1**

**46** represents number **3**

So, the numeric **1** differs from ASCII **'1'**.

**What is the purpose of getting a character's ASCII value?**

Assume you are now **18** years old. You'll be **19** in a year. As we all know, your age **varies** from year to year.

- Is it possible, however, for your phone number to change after a certain amount of time?

- - > No, of course we cannot.

Because these numbers do not undergo any changes, the computer keeps them in the form of **ASCII values**.

This also applies to your address, if you live at **29** XXXX street, Will it be changed to **35** XXXX street at any time?

Of course not, it's a **fixed** number. As a result, it is stored as an **ASCII value**.

Therefore, any **character** is represented by a **value** rather than the character itself.