Tentu, ini adalah rangkuman fundamental yang Anda perlukan untuk memahami dan bekerja dengan char (karakter) di bahasa C.

Pada dasarnya, **semua berawal dari ASCII**.

---

1. Konsep Inti: char adalah Angka

Hal terpenting yang harus dipahami adalah: char di C pada dasarnya adalah **integer (bilangan bulat) berukuran 1-byte**.

Komputer tidak menyimpan 'A', 'B', atau '#'. Komputer menyimpan angka yang mewakili karakter-karakter tersebut. Standar pemetaan yang paling umum adalah **ASCII** (American Standard Code for Information Interchange).

- Saat Anda menyimpan char c = 'A';, Anda sebenarnya menyimpan **angka 65** ke dalam variabel c.
- Saat Anda menyimpan char spasi = ' ';, Anda menyimpan **angka 32**.
- Saat Anda menyimpan char angka = '0';, Anda menyimpan **angka 48**.

Karena char adalah angka, Anda bahkan bisa melakukan operasi matematika dengannya:

C

```
char c = 'A'; // Nilai ASCII-nya 65

c = c + 1;    // Sekarang nilai c adalah 66

printf("%c", c); // Akan mencetak 'B'
```

---

2. Deklarasi char (Satu Karakter)

Untuk menyimpan *satu* karakter, Anda mendeklarasikannya menggunakan tipe char dan menginisialisasinya menggunakan **kutip tunggal** (').

C

```
char nilaiUjian = 'A';

char simbol = '%';

char digit = '7';
```

Penting: char digit = '7'; (karakter '7', ASCII 55) **sangat berbeda** dengan int angka = 7; (angka 7).

---

3. "String": Array dari char

Bahasa C tidak memiliki tipe data string bawaan seperti Python atau Java.

Di C, **string** adalah sebuah **array dari char** yang diakhiri dengan karakter khusus yang disebut **null terminator** ('\0').

Karakter '\0' (yang memiliki nilai ASCII 0) sangat penting; ini berfungsi sebagai penanda bagi fungsi-fungsi di C untuk mengetahui di mana sebuah string berakhir.

Deklarasi String:

Ada dua cara umum untuk membuat string:

1. Sebagai Array char (Paling Umum):

Saat Anda menggunakan kutip ganda ("), C secara otomatis membuat array char dan menambahkan '\0' di akhir.

C

// Cara 1: C mengurus ukurannya

char sapaan[] = "Halo";

// Cara 2: Anda menentukan ukurannya (harus cukup)

char nama[20] = "Budi";

Visualisasi "Halo" di memori:

['H', 'a', 'l', 'o', '\0']

(Membutuhkan 5 byte, meskipun panjangnya 4 karakter)

2. Sebagai Pointer char (Read-Only):

C

char *sapaan2 = "Dunia";

Ini membuat pointer ke lokasi memori *read-only* tempat "Dunia" disimpan. Anda biasanya tidak dapat mengubah isi string ini.

Perbedaan Kunci: ' vs "

- 'A' (kutip tunggal): Adalah satu char.
- "A" (kutip ganda): Adalah *string*, yang secara teknis merupakan array char berisi ['A', '\0'].

---

4. Input & Output (I/O)

Untuk membaca dan mencetak char dan string, Anda menggunakan *format specifier* yang berbeda di printf() dan scanf().

- %c : Untuk **satu char**.
- %s : Untuk **string** (array char).

C

#**include** <stdio.h>

```
int main() {
    // --- Bagian char (%c) ---
    char huruf;
    printf("Masukkan satu huruf: ");

    // Gunakan spasi sebelum %c untuk "memakan" karakter newline
    // dari input sebelumnya (ini adalah trik umum)
    scanf(" %c", &huruf);
    printf("Huruf yang Anda masukkan: %c\n", huruf);



    // --- Bagian string (%s) ---
    char kata[30]; // Siapkan buffer (tempat) berukuran 30 char
    printf("Masukkan satu kata: ");

    // Tidak perlu & untuk array saat menggunakan %s di scanf
    scanf("%s", kata);
    printf("Kata yang Anda masukkan: %s\n", kata);

    return 0;
}
```

---

5. Pustaka Standar yang Wajib Tahu

Anda tidak perlu "menemukan kembali roda". C menyediakan *library* (pustaka) untuk memanipulasi char dan string.

include <string.h> (Untuk String)

Karena string adalah array, Anda **tidak bisa** membandingkan atau menyalinnya menggunakan operator == atau =. Anda harus menggunakan fungsi dari string.h.

- strlen(str): Menghitung panjang string (berhenti di \0).
    - strlen("Halo") mengembalikan 4.
- strcpy(tujuan, sumber): Menyalin string sumber ke tujuan.

- strcat(tujuan, sumber): Menggabungkan string sumber ke akhir string tujuan.
- strcmp(str1, str2): Membandingkan dua string. Mengembalikan 0 jika keduanya sama.

include <ctype.h> (Untuk Char)

Pustaka ini sangat berguna untuk menguji atau mengubah *satu karakter*.

- isdigit(c): Benar jika c adalah digit ('0' - '9').
- isalpha(c): Benar jika c adalah huruf alfabet.
- islower(c) / isupper(c): Cek apakah huruf kecil atau besar.
- tolower(c) / toupper(c): Mengubah c menjadi huruf kecil atau besar.

Rangkuman Singkat

1. char adalah angka 1-byte (ASCII).
2. 'A' (kutip tunggal) adalah satu char.
3. "Halo" (kutip ganda) adalah *string* (array char).
4. String *harus* diakhiri dengan '\0' (null terminator).
5. Gunakan %c untuk char dan %s untuk string.
6. Gunakan <string.h> untuk operasi string (seperti strcmp, strcpy).
7. Gunakan <ctype.h> untuk memeriksa satu char (seperti isdigit, isalpha).

Apakah Anda ingin mencoba contoh program spesifik menggunakan char, seperti memvalidasi input pengguna atau membalikkan sebuah string?