C++ References :-

In C++, a reference is esentially an alias for another variable. It allows you to refer to an existing variable or object directly, without making a copy of it. You can think of it as a "nickname" for the original variable.

1. Reference Syntax :-

declared using & symbol in C++.

```
int a = 10;
int & ref = a; //ref is a reference to 'a'.
```

- Any changes made to ref will directy affect a, cuz ref is just another name for a.

2. References are not pointers

- A reference is not a pointer. While both can be used to refer to another variable, references behave more like aliases. They are easier to use cuz you don't have to dereference them like pointer.

- You cannot change a reference to refer to a different variable after it's been initialized.

```
int a = 20;
int b = 10;
int & ref = a; // ref is now an alias for a
ref = b;       // this changes the value of a to 10, but ref still
               // refers to a.
```

3. No null references :-

- Unlike pointers, references cannot be NULL. A reference must always be bound to an object (or variable). There is no concept of a null reference.

```
int a = 10;
int & ref = a;  // ref must refer to an existing variable.
// int &ref2;  // this would be an error since ref2 is not bound to anything.
```

4. References as function arguments :-

- Passing by reference allows a fn to modify the actual argument passed to it. This is efficient because the function does not make a copy of the argument (which would be the case if passed by value).

```
void increment(int & num){
        num++;  // directly modifies the original variable
}

int main(){
    int a = 5;
    increment(a);
    std::cout<<a;
}
```

- Passing by reference is commonly used in C++ for efficiency, [illegible] working with large objects or structures, as it

- Passing by reference is [illegible] especially when working with large objects or structures, as it avoids copying the data.

5. References as return values :-

- A function can return a reference to a variable. This can be useful when you want to modify a variable from outside the fn or chain operations.

```
int a=10;
int& getA(){
    return a; //returns a reference to 'a'
}

int main(){
    int& ref = getA();
    ref = 20;
    std::cout << a;
}
```