C++ Proqramlaşdırma Dili: Funksiyalardan (Altproqramlardan) İstifadə

C++ dilində funksiyalar (altproqramlar) çox vacib və güclü bir xüsusiyyətdir. Funksiyalar kodu təşkil etməyə, təkrar istifadəyə və oxunaqlılığı artırmağa kömək edir. Bu mühazirədə funksiyaların nə olduğunu, necə istifadə olunduğunu və hansı növlərinin mövcud olduğunu ətraflı şəkildə izah edəcəyik.

1. Funksiya Nədir?

Funksiya, müəyyən bir əməliyyatı yerinə yetirmək üçün yazılmış, təkrarlanan və ya ayrı-ayrı hissələrdə istifadə edilə bilən bir koddur. Funksiya daxilindəki əmrlər və ifadələr bir yerə toplanır, funksiyadan çağırıldığında bu əməliyyatlar yerinə yetirilir.

C++ dilində funksiyalar bir çox məqsəd üçün istifadə olunur. Bu məqsədlər arasında kodun sadələşdirilməsi, funksiyaların təkrarı və səhvlərin qarşısının alınması yer alır.

2. Funksiya Tərifi

Funksiya yaratmaq üçün əvvəlcə onun **tipi**, **adı**, **parametrləri** (əgər varsa) və **geri qaytarılacaq dəyəri** təyin olunur.

Funksiya tərifi aşağıdakı şəkildə olur:

cpp Copy Edit

```cpp
tip adi(parametr1, parametr2, ...) {
    // funksiyanın bədənindəki əmrlər
}
```

- **tip**: Funksiyanın geri qaytaracağı verilənlər tipidir. Məsələn, int, float, void və s.
- **adi**: Funksiyanın adı.
- **parametr1, parametr2, ...**: Funksiyaya verilən dəyərlər (əgər funksiyanız parametr qəbul edirsə).

- **funksiyanın bədəni**: Funksiyanın yerinə yetirəcəyi əməliyyatlar.

3. Funksiyanın Növü

Funksiyalar, geri qaytardıqları dəyərə və məqsədinə görə iki əsas növə ayrılır:

- **Geri qaytaran funksiyalar**: Bu funksiyalar müəyyən bir dəyər geri qaytarır. Məsələn, int və ya float tipində ola bilər. Bu funksiyalar əsasən hesablama və nəticə təqdim etmə məqsədilə istifadə olunur.
- **Geri qaytarmayan funksiyalar (void)**: Bu funksiyalar heç bir dəyər geri qaytarmır. Funksiya sadəcə əməliyyat yerinə yetirir. Məsələn, ekrana yazı yazdırmaq üçün istifadə edilən funksiyalar.

4. Funksiyanın Çağırılması

Funksiya tərif edildikdən sonra onu çağırmaq lazımdır. Funksiyanı çağırmaq üçün onun adını və parametrlərini yazırıq:

```cpp
adi(parametr1, parametr2, ...);
```

5. Funksiya Nümunələri

5.1. Geri Qaytaran Funksiya

Aşağıda topla adlı iki ədədi toplayan funksiya yaradılıb:

```cpp
#include <iostream>
using namespace std;

int topla(int a, int b) {
    return a + b;
}

int main() {
    int result = topla(3, 5);
    cout << "Toplam: " << result << endl;
    return 0;
}
```

İzah: Bu funksiyanın adı topladır və iki int tipində parametr qəbul edir. Funksiya bu iki ədədi toplayıb geri qaytarır. Burada funksiyanın geriyə qaytardığı nəticə result dəyişənində saxlanılır və ekrana yazdırılır.

5.2. Geri Qaytarmayan Funksiya

Aşağıda goster adlı sadə bir funksiya yaradılıb ki, bu funksiya sadəcə bir mesajı ekrana yazdırır:

cpp

Copy Edit

```cpp
#include <iostream>
using namespace std;

void goster() {
    cout << "Salam, bu geri qaytarmayan funksiyadır!" << endl;
}

int main() {
    goster();
    return 0;
}
```

İzah: Bu funksiyanın void qaytarma tipi var, yəni heç bir dəyər geri qaytarmır. Funksiya sadəcə ekrana bir mesaj yazdırır. Funksiya goster() main() funksiyasından çağırılır və mesaj ekranda görünür.

6. Parametrlər və Argumentlər

Funksiyalar parametr qəbul edərək xarici məlumatlarla işləyə bilər. Parametrlər funksiyanı çağıran zaman ona göndərilən verilənlərdir.

Məsələn:

```cpp
#include <iostream>  // Ekrana məlumat çıxarmaq və istifadəçidən məlumat almaq üçün lazım olan kitabxana
using namespace std; // Standart ad məkanından istifadə olunur

// Funksiya tərifi:
// Bu funksiya istifadəçinin adını parametr olaraq qəbul edir və onu salamlayır.
void salamla(string ad) {
    // Ekrana salamlaşma mesajı çıxarılır
    cout << "Salam, " << ad << "!" << endl;
}

// Əsas funksiya (programın başlanğıcı)
int main() {
    // 'salamla' funksiyası çağırılır və parametr kimi istifadəçi adı göndərilir
    salamla("Ramal Asgarov");  // Funksiya işləyəcək və ekrana "Salam, Ramal Asgarov!" çıxacaq

    return 0;  // Proqramın uğurla bitdiyini göstərir
}

```

İzah: Bu funksiyada ad adlı parametr qəbul edilir və ona göndərilən verilənlər istifadə olunur. main() funksiyasından salamla("Ramal Asgarov") çağırışında "Ramal Asgarov" dəyəri funksiyaya ötürülür.

6.1. Parametrlərin Dəyişdirilməsi

Funksiyalarda parametrin dəyərini dəyişdirmək mümkündür. Bunun üçün parametrin **referans**la ötürülməsi lazımdır.

```
1   #include <iostream>
2   using namespace std;
3
4   void deyişdir(int& a) {
5       a = a + 10;
6   }
7
8   int main() {
9       int num = 5;
10      deyişdir(num);
11      cout << "Yeni dəyər: " << num << endl;
12      return 0;
13  }
14
```

İzah: int& a yazaraq, a parametrini referansla göndəririk. Bu zaman funksiyadan sonra num dəyişkəninin dəyəri dəyişir. & işarəsi, parametrin dəyişdirilməsi üçün referansla ötürülməsini təmin edir.

7. Funksiya Overloading (Eyni Adla Bir neçə Funksiya)

C++ dilində eyni adda, amma fərqli parametr tipləri və ya sayları olan bir neçə funksiya təyin etmək mümkündür. Bu xüsusiyyətə **funksiya overloading** deyilir.

```cpp
#include <iostream>
using namespace std;

int topla(int a, int b) {
    return a + b;
}

float topla(float a, float b) {
    return a + b;
}

int main() {
    cout << "Int toplama: " << topla(3, 4) << endl;
    cout << "Float toplama: " << topla(3.5f, 4.2f) << endl;
    return 0;
}
```

İzah: Həm int həm də float tipində parametr qəbul edən iki ayrı topla funksiyası yaradılıb. Bu zaman funksiyanı çağıran zaman verilənlərin tipinə uyğun olan versiya seçilir.

8. Rekursiya

Funksiya özünü çağırarsa, bu **rekursiya** adlanır. Rekursiya ilə işləyən funksiyalar, müəyyən şərtlə özünü təkrarlayır və nəticədə verilən məsələni həll edir.

Məsələn, faktorial hesablayan rekursiv funksiya:

```cpp
#include <iostream>
using namespace std;

int faktorial(int n) {
    if (n == 0)
        return 1;
    else
        return n * faktorial(n - 1);
}

int main() {
    cout << "5! = " << faktorial(5) << endl;
    return 0;
}
```

İzah: faktorial funksiyası özünü çağırır və hər dəfə n-in dəyərini bir vahid azaldır. Bu şəkildə faktorial dəyəri tapılır. Rekursiv funksiyanın əsas xüsusiyyəti, müəyyən bir təkrarlanma şərtini təmin etdikdən sonra özünü çağırmasıdır. Burada, n == 0 olduqda, funksiyanın nəticəsi 1 olur və rekursiya dayanar.

9. Funksiya Tərifinin və Çağırılmasının Yeri

C++-da funksiyalar **başlıq faylında** tərif oluna və **mənbə faylında** çağırıla bilər. Bu, kodun daha təmiz və təşkilatlı olmasına kömək edir.

9.1. Funksiya Deklarasiyası və Tərifi

Bəzən, funksiyanı istifadə etmədən əvvəl onu **deklarasiya** etmək lazımdır. Bu, xüsusilə böyük proqramlarda vacibdir.

```cpp
#include <iostream>
using namespace std;

void goster();  // Deklarasiya

int main() {
    goster();
    return 0;
}

void goster() {  // Tərif
    cout << "Funksiya işə düşdü!" << endl;
}

```

İzah: Funksiya əvvəlcə yalnız adı və parametrləri ilə deklarasiya edilir, sonra isə əsas kodda təyin olunur.

10. Nəticə

C++ dilində funksiyalar proqramın daha səliqəli, oxunaqlı və təkrar istifadə edilə bilən olmasını təmin edir. Funksiyalar müxtəlif növləri ilə çox yönlüdür və mürəkkəb məsələlərin həllində əvəzsizdir. Funksiyaların istifadəsi, kodun təkrarlanmamasını və səhvlərin qarşısını alır.

Funksiya yaratmaq, çağırmaq, parametr qəbul etmək və nəticə geri qaytarmaq kimi əsas prinsipləri yaxşı başa düşmək, C++ proqramlaşdırma dilində mükəmməl nəticələr əldə etməyinizə kömək edəcək.