4.Ders : C Unions

Bu derste, C programlamadaki **unions** hakkında bilgi edineceksiniz. Daha spesifik olarak, unionsların nasıl oluşturulacağı, üyelerine nasıl erişileceği ve unions ile yapılar arasındaki farkların nasıl öğrenileceğini göreceksiniz.

Bir bellek alanına, farkı zamanlarda, farklı değişkenlerden istenen birisinin değerini girebilmek gerekir. C dili buna bir çözüm getirmek için, bize **union** adlı bir veri türü daha sunmaktadır.

Union, bileşenlerine ortak bir bellek alanını kullandıran bir veri türüdür.

Bir union nasıl tanımlanır?

Union'ları tanımlamak için union anahtar sözcüğünü kullanıyoruz. İşte bir örnek:

```
union car
{
  char name[50];
  int price;
};
```

Union değişkenleri oluşturmak.
Bir Union tanımlandığında, kullanıcı tanımlı bir tür oluşturur. Ancak, hiçbir hafıza tahsis edilmemiştir. Belirli

bir birleşim türüne bellek ayırmak ve onunla çalışmak için değişkenler oluşturmamız gerekir.

İşte Union değişkenlerini böyle yaratıyoruz :

```
union car
{
  char name[50];
  int price;
};
int main()
{
  union car car1, car2, *car3;
  return 0;
}
```

başka bir yolu:

```
union car
{
  char name[50];
  int price;
} car1, car2, *car3;
```

Her iki durumda da, car1, car2 ve union car tipinde bir **union pointer car3** birleşim değişkenleri oluşturulur.

Union erişim üyeleri
Bir Union üyelerine erişmek için operatör kullanıyoruz. İşaretçi değişkenlerine erişmek için -> operatörünü de kullanırız.

Yukarıdaki örnekte,

Car1 fiyatına erişmek için car1.price kullanılır.
Car3 kullanarak fiyatlara erişmek için, (* car3) .price veya car3-> fiyatları kullanılabilir.

Union ve yapılar arasındaki farklar

Union ve yapılar arasındaki farkı göstermek için bir örnek alalım:

```
#include <stdio.h>
union unionJob
{
   //defining a union
   char name[32];
   float salary;
   int workerNo;
} uJob;
struct structJob
{
   char name[32];
   float salary;
   int workerNo;
} sJob;
int main()
{
   printf("size of union = %d bytes", sizeof(uJob));
   printf("\nsize of structure = %d bytes", sizeof(sJob));
   return 0;
}
```

Çıktısı :

```
size of union = 32
```

```
size of structure = 40
```

Bir seferde yalnızca bir Union üyesine erişilebilir.
Bir yapının tüm üyelerine bir kerede tüm üyelere yeterli bellek tahsis edildiğinde erişebilirsiniz. Ancak Union'larda durum böyle değil. Bir kerede yalnızca bir birliğin bir üyesine erişebilirsiniz. Bir örnek görelim :

```
#include <stdio.h>
union Job
{
   float salary;
   int workerNo;
} j;
int main()
{
   j.salary = 12.3;
   j.workerNo = 100;
   printf("Salary = %.1f\n", j.salary);
   printf("Number of workers = %d", j.workerNo);
   return 0;
}
```

Çıktısı :

```
Salary = 0.0
Number of workers = 100
```