به نام خدا

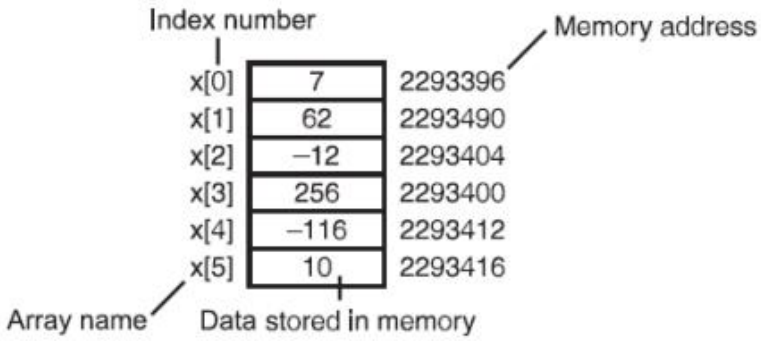
دانشکده‌ی مهندسی برق و کامپیوتر دانشکده فنی دانشگاه تهران

مبانی کامپیوتر و برنامه نویسی

نیمسال اول	عنوان:	اساتید:
1402-03	آرایه‌ها	دکترمرادی، دکتر هاشمی

در این جلسه شما با آرایه‌ها[1] و نوع خاصی از آن‌ها یعنی رشته‌ها[2] آشنا خواهید شد.

تعریف آرایه: تعدادی داده‌های پشت سر هم در حافظه که همگی از یک نوع[3] هستند.



نکته 1: شماره اندیس آرایه به طول n از 0 شروع و تا n-1 ادامه دارد.

برای تعریف و مقداردهی آرایه‌ها در زبان c روش‌های متعددی وجود دارد. در کد زیر تعدادی از این روش‌ها ذکر شده‌اند:

```
√ int a[] = { 5, 3, 2 };
√ int b[3];
√ int c[3] = { 6, 5 };
√ b[0] = 2;

√ int d[3];
× d = { 4, 5, 6 };
√ int e[3];
× e[3] = 4;
```

نکته 2: مقداردهی آرایه با استفاده از {} تنها هنگام تعریف آرایه مجاز است.

نکته 3: طول آرایه‌ها باید در زمان compile تعیین شود. بدین معنا که طول آرایه در زمان اجرای برنامه نمی‌تواند تعریف یا تغییر داده شود. با توجه به این موضوع طول آرایه باید **مقدار ثابتی** داشته باشد. می‌توان با دستور #define مقدار ثابتی را تعریف نمود.

---

[1] Arrays
[2] Strings
[3] Type

1- انجام دهید!

یک پروژه‌ی جدید ایجاد کرده و کد زیر را در آن کامپایل کنید.

```
#include<stdio.h>
int main(){
    int num = 5;
    int arr[num] = {1, 2, 3, 5};
    return 0;
}
```

- به compile error ایجاد شده توجه کنید. چرا همچین خطایی رخ می‌دهد؟ علت را برای دستیاران آموزشی توضیح دهید.(قسمت 1)
- حال نوع متغیر num را از `int` به `const int` تغییر دهید. آیا خطای کامپایل برطرف شد؟ چرا؟ می‌توانید علت را در این لینک مطالعه کنید.
- حال قطعه کد زیر را اجرا کنید.

```
#include<stdio.h>
#define NUM 5
int main(){
    int arr[NUM] = {1, 2, 3, 5};
    return 0;
}
```

چه اتفاقی افتاد؟ چرا با وجود اینکه در تعریف آرایه از NUM استفاده کردیم، برنامه بدون خطا کامپایل و اجرا می‌شود؟ علت را برای دستیاران آموزشی توضیح دهید.(قسمت 2)

- حال در برنامه‌ی زیر، قسمت‌های خالی را طوری تکمیل کنید که برنامه با استفاده از for و دستور scanf، 5 عدد از کاربر دریافت نموده و در خانه‌های آرایه ذخیره کند. سپس با استفاده از دستور printf مقادیر وارد شده را به ترتیب معکوس در خروجی نمایش دهد.

```
#include<stdio.h>
int main(){
    int arr[5];
    for(int i= ...;...;i++){
        scanf("%d", &arr[...]);
    }
    for(int j= ...;...;...){
        printf("%d\n", ... );
    }
    return 0;
}
```

قسمت 3: نتیجه را به دستیاران آموزشی نشان دهید.

توجه: همانطور که در برنامه‌ی بالا می‌بینید، برای دریافت مقدار یک خانه از آرایه، از علامت & استفاده شده است. درواقع آدرس آن خانه از آرایه را فراخوانده و مقدار ورودی را در آن آدرس ذخیره می‌کنیم.

2- انجام دهید!

فرض کنید یک آرایه داریم و می‌خواهیم محتوای آن را در یک آرایه‌ی دیگر، کپی کنیم. برنامه زیر به همین منظور نوشته شده است. این برنامه را در یک پروژه‌ی جدید کامپایل کنید.

```
#include<stdio.h>
int main(){
    int arr[5]={1,2,3,4,5};
    int arr_copy[5];
    arr_copy=arr;
    return 0;
}
```

- به compile error ایجاد شده توجه کنید. چرا همچین خطایی رخ می‌دهد؟ علت را برای دستیاران آموزشی توضیح دهید.(قسمت 4)
- حال برنامه‌ی زیر را طوری تکمیل کنید که درایه‌های آرایه‌ی arr، در آرایه‌ی arr_copy، کپی شود.

```
#include<stdio.h>
int main(){
    int arr[5]={1,2,3,4,5};
    int arr_copy[5];
    for(...){
        ...;
    }
    return 0;
}
```

قسمت 5: نتیجه را به دستیاران آموزشی نشان دهید.

3- انجام دهید!

همانطور که می‌دانید رشته (string)، آرایه‌ای از کاراکترها می‌باشد. دو برنامه‌ی زیر را در یک پروژه‌ی جدید اجرا کنید.

```
#include<stdio.h>
int main(){
    char str1[5]="hello";
    printf("string 1= %s\n", str1);
    char str2[6]="hello";
    printf("string 2= %s\n", str2);
    return 0;
}
```

```
#include<stdio.h>
int main(){
    char str1[]={'h','e','l','l','o'};
    printf("string 1= %s\n", str1);
    char str2[]={'h','e','l','l','o','\0'};
    printf("string 2= %s\n", str2);
    return 0;
}
```

- چرا در هردو برنامه خروجی آرایه‌های str1 و str2 با هم متفاوت است؟ علت را برای دستیاران آموزشی توضیح دهید.(قسمت 6)

راهنمایی: در مورد NULL character در این لینک مطالعه کنید!

توجه: همانطور که دیدید یک راه برای نمایش رشته‌ها استفاده از %s است. راه دیگر آنست که با استفاده از یک حلقه‌ی for، تمام اعضای رشته را یکی‌یکی توسط %c در خروجی نمایش دهیم.

4- انجام دهید!

اکنون برنامه‌ی زیر را در یک پروژه‌ی جدید اجرا کرده و عبارت "hello world" را به عنوان ورودی وارد کنید.

```
#include<stdio.h>
int main(){
    char str[50];
    scanf("%s", str);
    printf("%s\n", str);
    return 0;
}
```

- آیا عبارت نمایش داده شده در خروجی، با عبارت ورودی یکسان است؟ علت را برای دستیاران آموزشی توضیح دهید.(قسمت 7)

برای پیشگیری از این مشکل، می‌توان از تابع gets() استفاده کرد. در این تابع، فقط کلید Enter انتهای رشته را مشخص می‌کند. لذا رشته می‌تواند حاوی فاصله (space) و یا Tab باشد.

حال در برنامه بالا، عبارت scanf را حذف کرده و آنرا با gets(str) جایگزین کنید. اینک دوباره نتیجه را بررسی نمایید.

- حال برای آشنایی با تابع puts() که برای چاپ رشته‌ها استفاده می‌شود، برنامه‌ی زیر را اجرا کنید. چه تفاوتی میان توابع printf و puts() وجود دارد؟

```
#include<stdio.h>
int main(){
    char str[]="hello world";
    printf("*****\n");
    puts(str);
    printf("*****\n");
    printf("%s", str);
    printf("*****\n");
    return 0;
}
```

قسمت 8: نتیجه را برای دستیاران آموزشی توضیح دهید.

✓ **توصیه می‌شود در مورد توابع getch ، getche ، getchar ، putch و putchar و کاربردهای آنها تحقیق و مطالعه نمایید. برای این کار می‌توانید به این لینک مراجعه کنید.**

پیش از انجام قسمت بعد، ابتدا برخی توابع پرکاربرد کتابخانه string.h را بررسی می‌کنیم:

- **تابع strcpy()**: این تابع برای کپی کردن رشته‌ای در رشته دیگر و یا انتساب رشته‌ای به رشته‌ای دیگر استفاده می‌شود و به صورت زیر به‌کار می‌رود:

```
strcpy(str1 , str2);
```

با اجرای این دستور، آنچه که در str2 است، در str1 کپی می‌شود.

- **تابع strcmp()**: این تابع برای مقایسه رشته‌ها استفاده می‌شود و به صورت زیر به‌کار می‌رود:

```
strcmp(str1 , str2);
```

حاصل کار این تابع یک عدد است که مقدار آن بیانگر وضعیت دو رشته نسبت به هم است. اگر عدد برگردانده شده توسط این تابع برابر صفر باشد، این دو رشته با هم مساویند. در غیراین صورت، این دو رشته با یکدیگر مساوی نمی‌باشند. برای کسب اطلاعات بیشتر در مورد این تابع و معنی مقادیر برگردانده شده توسط آن، می‌توانید به این لینک مراجعه کنید.

- **تابع strcat()**: با استفاده از این تابع می‌توان دو رشته را با هم الحاق کرد. این تابع به صورت زیر به‌کار می‌رود:

```
strcat(str1 , str2);
```

با این دستور، str2 در انتهای str1 قرار می‌گیرد. چنانچه طول رشته str1 طوری باشد که گنجایش str2 را نداشته باشد، بقیه رشته str2 در ادامه رشته str1 قرار می‌گیرد و در نتیجه چنانچه متغیرهایی بعد از str1 وجود داشته باشند، محتویات آنها از بین می‌رود.[۴]

5- انجام دهید!

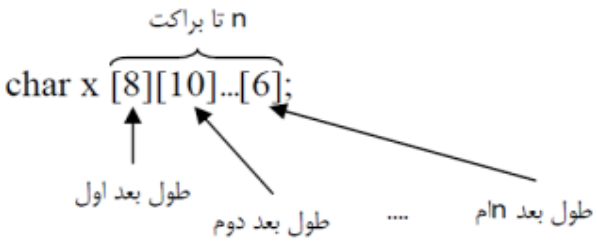
در این قسمت می‌خواهیم برنامه‌ای بنویسیم که رشته‌ای را دریافت کند و در صورتی که عبارت "hello world" وارد شده باشد، پیام "YES!" و در غیر این صورت، پیام "NO!" در خروجی نمایش داده شود. کد زیر را بدین منظور تکمیل کنید و نتیجه را به دستیاران آموزشی نشان دهید.(قسمت 9)

```
#include<stdio.h>
#include<...>
int main(){
    char str1[]="hello world";
    ...; //define str2
    ...; //input str2
    if(...){
        printf("YES!\n");
    }else{
        printf("NO!\n");
    }
    return 0;
}
```

[۴] برگرفته از کتاب برنامه نویسی به زبان c، دکتر جعفرنژاد قمی

آرایه‌های چند بعدی:

در زبان C می‌توان آرایه‌هایی با بیش از یک بعد نیز تعریف و استفاده کرد. نحوه‌ی تعریف یک آرایه از نوع کاراکتر با n بعد به صورت زیر است:



	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

دقت کنید: حافظه‌ی کامپیوتر مانند یک آرایه‌ی یک بعدی است. لذا برای شبیه سازی آرایه‌هایی با ابعاد بیش‌تر، سطرهای آن را پشت سر هم قرار می‌دهد و با استفاده از اشاره‌گر[5] به آن‌ها دسترسی پیدا می‌کند.

به عنوان مثال، برای آرایه‌ای دوبعدی از جنس int داریم:

```
int a[3][4]={{11,12,14,15},{21,22,23,24}, {13,31,32,33};
```

همچنین می‌توانید آرایه‌ی فوق را به این شکل نیز تعریف کنید:

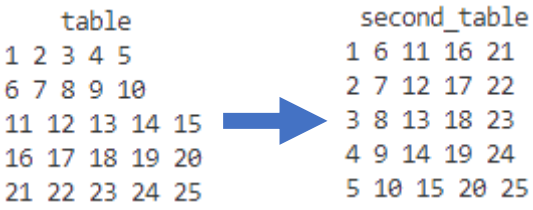
```
int a[3][4]={11,12,14,15,21,22,23,24,13,31,32,33};
```

دقت کنید: همانند آرایه‌های یک بعدی شما فقط هنگام تعریف یک آرایه‌ی چند بعدی می‌توانید آن را به صورت فوق مقداردهی کنید.

[5] Pointer

6- انجام دهید!

برنامه زیر را طوری تکمیل کنید که سطر اول آرایه‌ی دوبعدی table در ستون اول second_table و سطر دوم در ستون دوم و ... ذخیره گردد.(به تصویر زیر توجه کنید.) سپس ماتریس second_table را در خروجی چاپ کنید.




```c
#include<stdio.h>
#define SIZE 5
int main(){
    int i,j;
    int table[SIZE][SIZE]= {{1,2,3,4,5},
                            {6,7,8,9,10},
                            {11,12,13,14,15},
                            {16,17,18,19,20},
                            {21,22,23,24,25}};
    int second_table[SIZE][SIZE];
    for(int i=0;i<SIZE;i++){
        for(int j=0; j<SIZE;j++){
            ...;
        }
    }// transposing
    for(int i=0;i<SIZE;i++){
        for(int j=0;j<SIZE;j++){
            printf("%d ", ...);
        }
        printf("\n");
    }// printing the result
    return 0;
}
```

قسمت 10: نتیجه را به دستیاران آموزشی نشان دهید.

موفق باشید.

تهیه و تنظیم: امیرمرتضی رضائی