به نام خدا

طراحی زبان‌های برنامه‌سازی

مرجان عظیمی
Azimi.marjan@gmail.com

فصل ششم

بسته بندی

ساختمان داده‌ها

اشیای داده ساختاری و انواع داده

- شی داده ای که مرکب از اشیای داده دیگری است ساختمان داده نام دارد.
- بسیاری از مفاهیم و اصول مربوط به ساختمان داده‌ها در زبان‌های برنامه‌سازی مشابه اشیای داده اولیه است.

ساختمان داده ها (ادامه)

مشخصات انواع ساختمان داده

صفات اصلی مشخص کننده ساختمان داده:

- تعداد عناصر
- نوع هر عنصر
- اسامی برای انتخاب عناصر
- حداکثر تعداد عناصر
- سازمان عناصر

ساختمان داده‌ها (ادامه)

مشخصات انواع ساختمان داده (ادامه)

عملیات در ساختمان داده‌ها

دسته‌های دیگری از عملیات از اهمیت ویژه ای برخوردارند:

- عملیات انتخاب عناصر
- عملیات بر روی کل ساختمان
- درج و حذف عناصر
- ایجاد و حذف ساختمان داده‌ها

ساختمان داده ها (ادامه)

پیاده سازی انواع ساختمان داده ها

دو موضوع دیگر که انتخاب نمایش حافظه را تحت تاثیر قرار می‌دهد:

- انتخاب کارآمد عنصر از ساختمان
- مدیریت حافظه کارآمد برای پیاده سازی زبان

ساختمان داده ها (ادامه)

پیاده سازی انواع ساختمان داده ها (ادامه)

نمایش های حافظه

- حافظه ای برای عناصر ساختمان داده
- توصیفگر اختیاری آنها

دو نمایش اصلی:

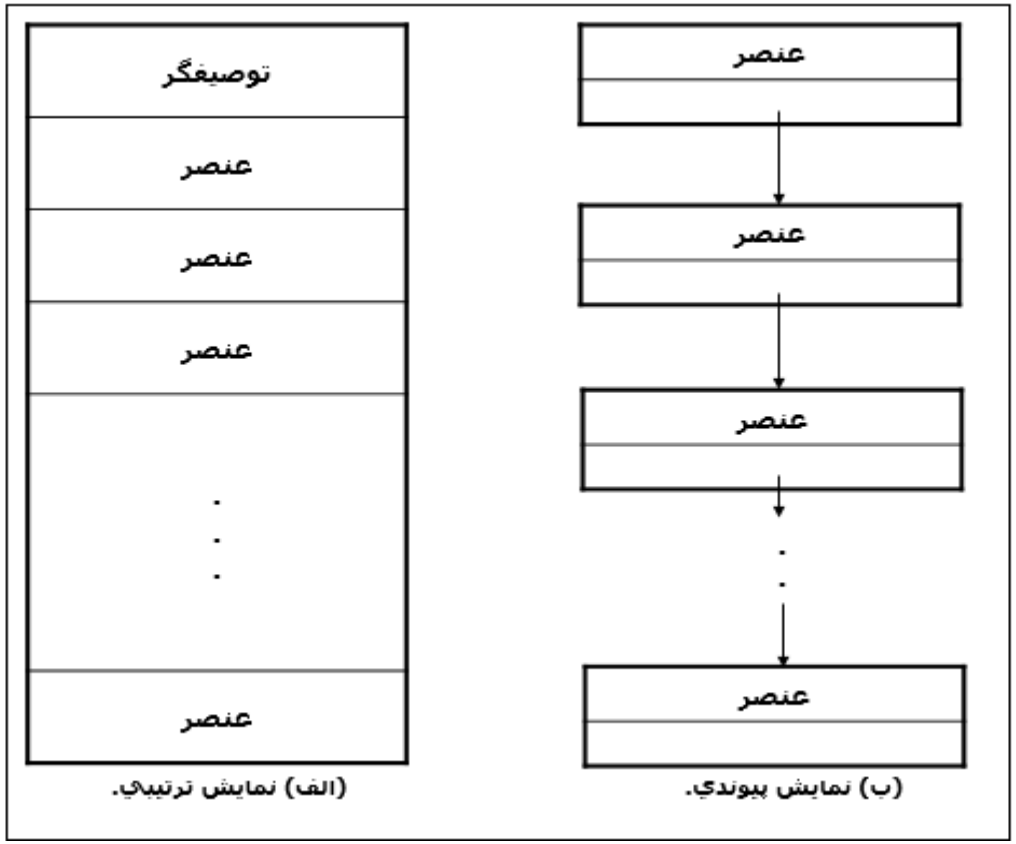
- نمایش ترتیبی
- نمایش پیوندی

ساختمان داده ها (ادامه)

- در نمایش ترتیبی، ساختمان داده در بلوک پیوسته ای از حافظه ذخیره می‌شود که شامل توصیفگر و عناصر ساختمان داده است.

- در نمایش پیوندی، ساختمان داده در چندین بلوک حافظه قرار می گیرد که معمولاً این بلوک‌ها پیوسته نیستند. ساختمان داده‌های طول متغیری مثل لیست‌ها از نمایش حافظه پیوندی استفاده می کنند.

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

پیاده سازی انواع ساختمان داده ها (ادامه)

پیاده‌سازی عملیات ساختمان داده‌ها

- انتخاب عناصر ساختمان داده مهمترین مسئله در پیاده‌سازی آن است.
- کارآمد بودن عملیات انتخاب تصادفی و انتخاب ترتیبی ضروری است.

ساختمان داده ها (ادامه)

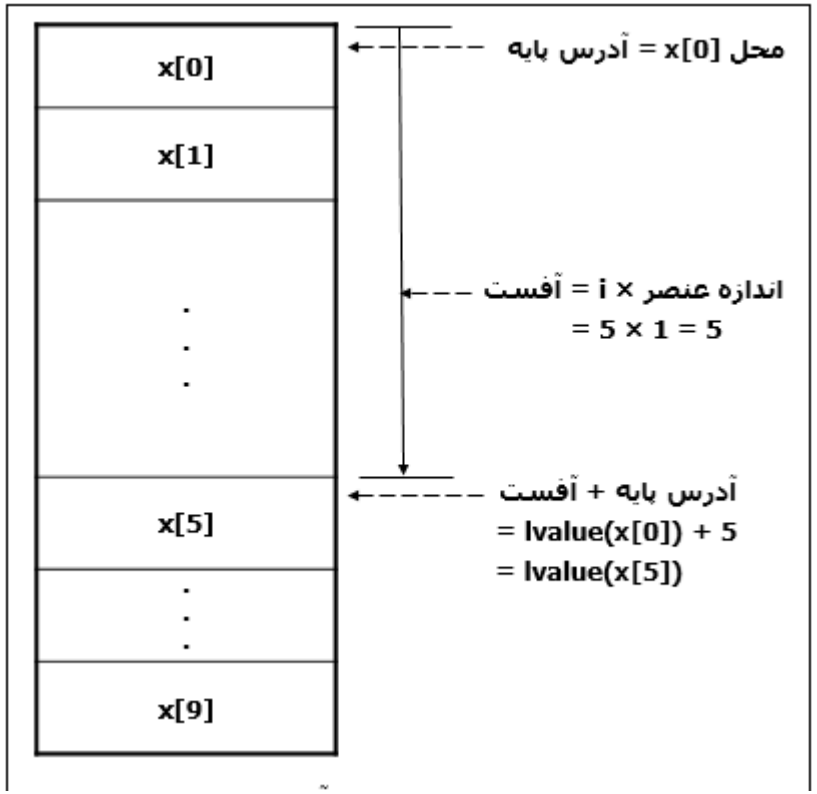
پیاده سازی عملیات ساختمان داده ها (ادامه)

- نمایش ترتیبی: در انتخاب تصادفی یک آدرس پایه – آفست باید با استفاده از فرمول دستیابی محاسبه شود.

در ساختار همگن انتخاب دنباله ای از عناصر می تواند به صورت زیر انجام شود:

- برای دستیابی به اولین عنصر دنباله از محاسبه آدرس پایه – آفست استفاده کنید.
- برای دستیابی به عنصر بعدی دنباله اندازه عنصر فعلی را به موقعیت عنصر فعلی اضافه کنید.

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

پیاده سازی عملیات ساختمان داده ها (ادامه)

نمایش پیوندی:

- برای انتخاب باید زنجیره‌ای از اشاره گرها را از اولین بلوک موجود در ساختار تا عنصر موردنظر دنبال کرد.
- برای انتخاب دنباله ای از مولفه ها باید اولین عنصر را انتخاب و سپس اشاره گر پیوندی را تا عنصر مورد نظر دنبال کرد.

ساختمان داده ها (ادامه)

پیاده سازی انواع ساختمان داده ها (ادامه)

مدیریت حافظه و ساختمان داده‌ها

- طول عمر هر شی داده با انقیاد شی به محلی از حافظه شروع می شود.

به علت تاثیر متقابل بین طول عمر شی داده و مسیرهای دستیابی دو مسئله مهم در مدیریت حافظه به وجود می آید:

- زباله
- ارجاع‌های معلق

ساختمان داده ها (ادامه)

اعلانها و کنترل نوع برای ساختمان داده‌ها

- مثل انواع داده اولیه است
- ساختمان داده ها پیچیده ترند زیرا صفات بیشتری باید مشخص شوند.
- دو مسئله در این مورد وجود دارد:
  - وجود مولفه انتخابی: ممکن است پارامترهای عملیات انتخاب درست باشد، ولی عنصر مورد نظر موجود نباشد.
  - نوع عنصر انتخابی: کنترل نوع ایستا تضمین می‌کند که اگر عنصر وجود دارد، نوع آن درست است.

ساختمان داده ها (ادامه)

بردارها و آرایه ها

- متداولترین ساختمان داده‌ها در زبان‌های برنامه‌سازی‌اند.
- بردار ساختمان مرکب از تعداد ثابتی از عناصر همنوع است که به صورت یک دنباله خطی سازمان‌دهی شده است.
- برای دستیابی به عناصر بردار از اندیس استفاده می شود.
- به بردار آرایه یک بعدی نیز گفته می‌شود.

ساختمان داده ها (ادامه)

بردارها و آرایه ها (ادامه)

بردارها:

- تعداد عناصر
- نوع هر عنصر
- اندیس برای انتخاب هر عنصر

ساختمان داده ها (ادامه)

بردارها و آرایه ها (ادامه)

عملیات بر روی بردارها:

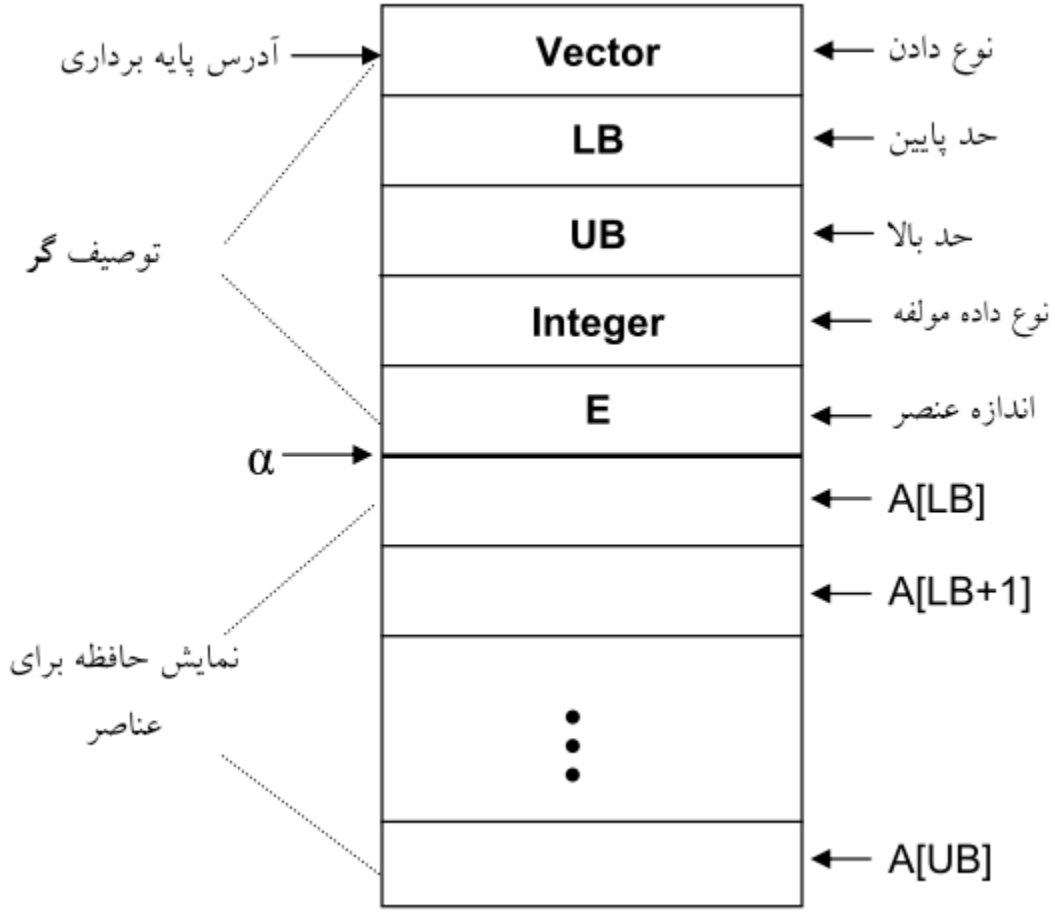
- عملیاتی که عنصری را از برداری انتخاب می کند اندیس گذاری نام دارد.
- ایجاد و حذف بردارها
- عملیات روی کل بردار

ساختمان داده ها (ادامه)

بردارها و آرایه ها (ادامه)

نمایش حافظه: نمایش حافظه بردار به صورت ترتیبی است. هر بردار توصیفگری در زمان ترجمه دارد که برای کنترل نوع، نمایش حافظه و محاسبه فرمول دستیابی (تابع دستیابی) به کار می‌رود.

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

حد پایین = Lb

حد بالا = ub

طول عنصر =E

اگر اولین عنصر بردار در محل α باشد، آدرس x[i]که آن را با address(x[i])نمایش می‌دهیم، به صورت زیر محاسبه می شود:

$$address(x[i]) = \alpha + (i - lb) * e = (\alpha - lb * e) + i * e$$

ساختمان داده ها (ادامه)

به جای ذخیره آدرس اولین عنصر بردار در توصیفگر، می توان آدرس دیگری را به نام مبدأ مجازی ذخیره نمود که مقدار ثابتی است و براساس آن تابع دستیابی را ساخت. می دانیم که:

$$address(x[i]) = \alpha + (i - Lb) * e$$
$$= (\alpha - Lb * e) + i * e$$
$$= k + i * e$$

اکنون با توجه به تابع دستیابی فوق، آدرس x[0] را محاسبه می کنیم:

$$address(x[0]) = k + 0 * e$$
$$= k$$
$$= vo$$

ساختمان داده ها (ادامه)

تخصیص حافظه برای بردار: حافظه ای برای nعنصر به طول eو توصیفگری به اندازه dتخصیص دهید. یعنی n * e + dمحل حافظه را ایجاد کنید.

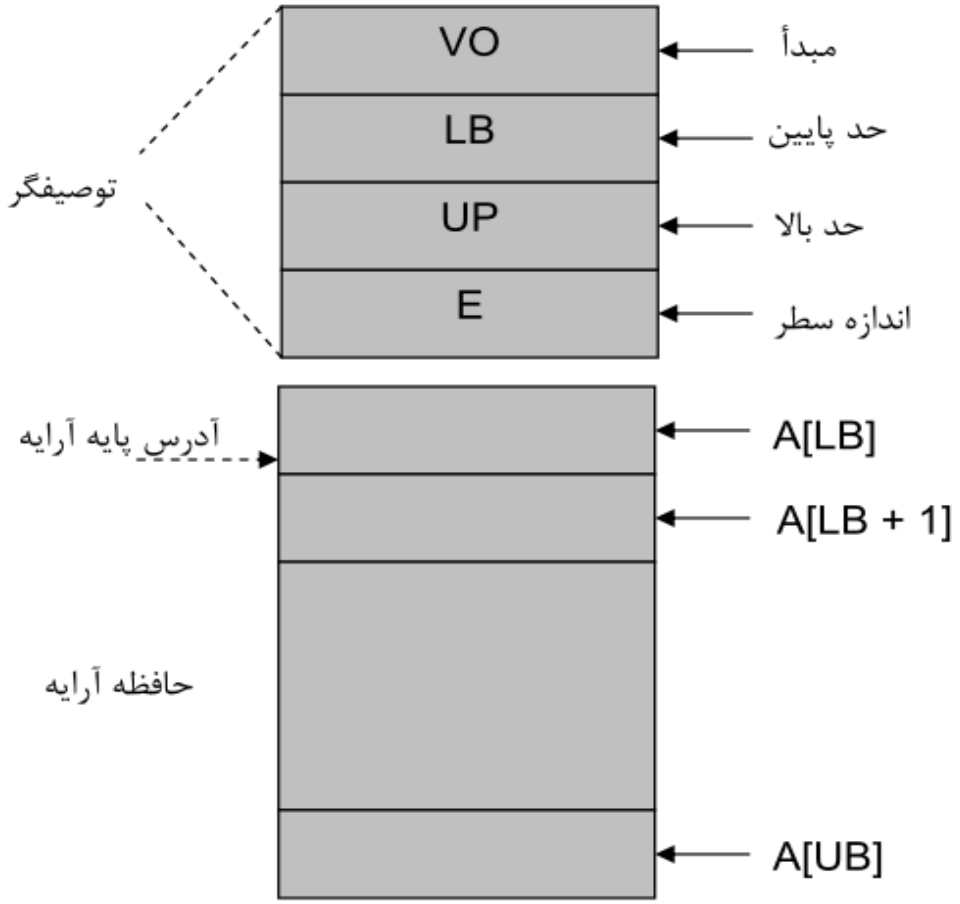
مبدأ مجازی را به صورت vo = α – lb * eمحاسبه کنید.

دستیابی به عنصری از بردار: آدرس هر عنصر x[i]را به صورت زیر محاسبه کنید:

address(x[i]) = vo + i * e

در این صورت، به جای قرار دادن اولین عنصر در توصیفگر، می توانیم مبدأ مجازی را قرار دهیم.

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

بردارها و آرایه ها (ادامه)

آرایه های چند بعدی

- مشخصات و نحو: تفاوت آرایه چند بعدی و بردار در بازه اندیس هر بعد است.
- پیاده سازی: می توان آن را برداری از بردارها در نظر گرفت .

ساختمان داده ها (ادامه)

به عنوان مثال، ماتریس X را در نظر بگیرید:

$$x = \begin{bmatrix} 3 & 4 & 7 \\ 6 & 2 & 5 \\ 1 & 3 & 8 \end{bmatrix}$$

در نمایش سطری عناصر آرایه X به ترتیب زیر ذخیره می‌شوند.

3 , 4 , 7 (سطراول) 6 , 2 , 5 (سطردوم) 1 , 3 , 8 (سطرسوم)

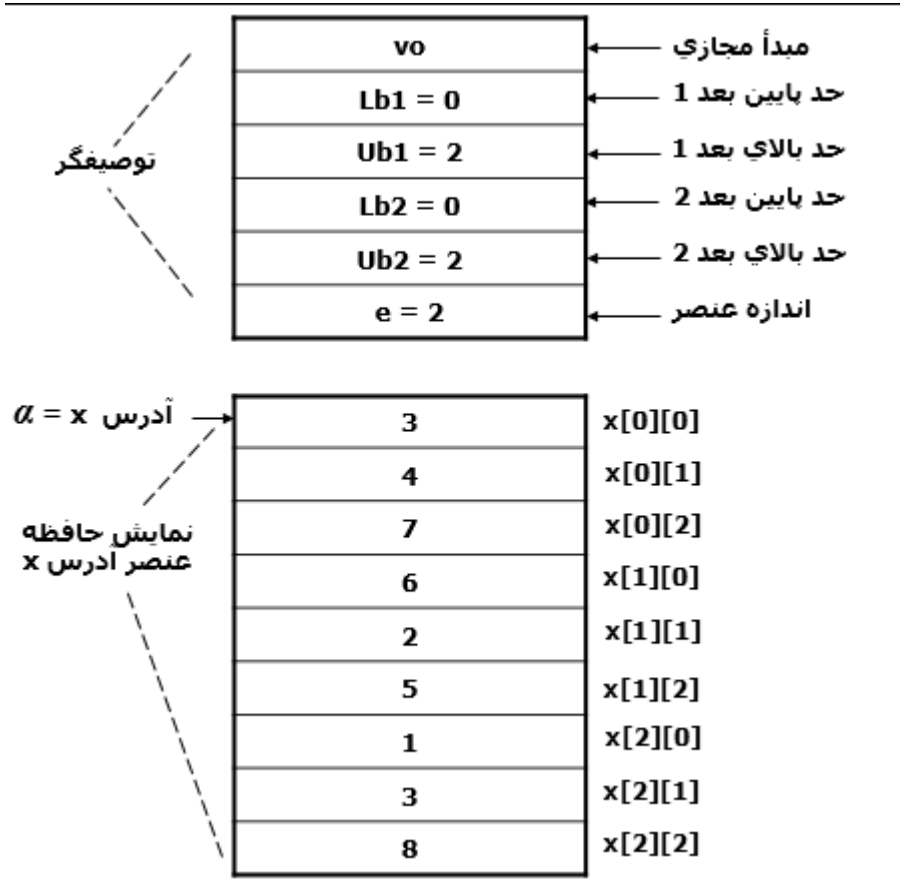
ساختمان داده ها (ادامه)

در نمایش ستونی، ابتدا ستون اول، سپس ستون دوم،... و سپس ستون آخر ذخیره می شوند. بنابراین، ماتریس xدر نمایش ستونی به صورت زیر ذخیره خواهد شد:

ستون اول	ستون دوم	ستون سوم
3 , 6 , 1	4 , 2 , 3	7 , 5 , 8

نمایش حافظه: نمایش حافظه آرایه دوبعدی، مانند بردار، شامل حافظه هایی برای توصیفگر و عناصر آرایه است. اگر مبدأ مجازی در توصیفگر ذخیره شود، توصیفگر و عناصر آرایه می توانند در حافظه پیوسته قرار نداشته باشند.

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

آدرس اولین عنصر آرایه = α

$$s = \underbrace{(ub2 - lb2 + 1) * e}_{\text{اندازه هرسطر}}$$

مبدأ مجازي = v

$$address(x[i][j]) = \alpha + (i - lb1) * s + (j - lb2) * e$$

$$= \underbrace{\alpha - lb1 * s - lb2 * e}_{\text{مقدارثابت}} + i * s + j * e$$

بنابراین داریم (i و j را صفر درنظر گرفتیم):

$$vo = \alpha - Lb1 * s - Lb2 * e$$

خواهیم داشت:

$$address(x[i][j]) = vo + i * s + j * e$$

ساختمان داده ها (ادامه)

رکوردها

- مشخصات و نحو: ساختمان داده های خطی با طول ثابت هستند اما رکوردها از دو جهت متفاوتند:
  - عناصر رکورد ممکن است ناهمگن و از انواع مختلفی باشند.
  - عناصر رکورد دارای نام هستند.
- پیاده سازی: نمایش حافظه برای رکورد شامل یک بلوک از حافظه است که عناصر درآن به ترتیب ذخیره می شود.

ساختمان داده ها (ادامه)

id	age	ave	name
100	18	15.5	Ali

ساختمان داده ها (ادامه)

اگر رکورد را R بناميم، از فرمول دستيابي زير براي يافتن آدرس عنصر i استفاده مي كنيم:

$$address(R.i) = \alpha + \sum_{j=1}^{i-1} (sizeof\ R.j)$$

α آدرس شروع بلوك حافظه رکورد و R.j عنصر زأم است. براي رسيدن به آدرس يك فيلد، بايد اندازه فيلدهاي قبلي با هم جمع شوند، اگر f_i را آفست عنصر i بناميم، در زمان ترجمه مي توان آن را حساب کرد و در زمان اجرا فقط لازم است آدرس پايه به اين آفست اضافه گردد:

$$address(R.i) = \alpha + f_i$$

ساختمان داده ها (ادامه)

رکوردها

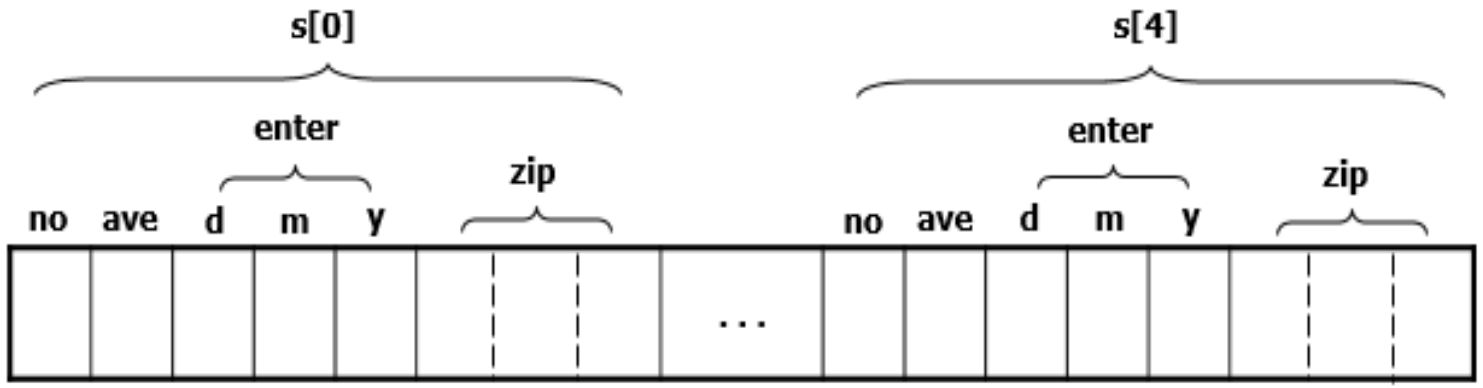
رکوردها و آرایه هایی با عناصر ساختاری

- عناصری از دو نوع مختلف با عناصری از انواع داده ترکیب شوند.
- انتخاب عناصر مستلزم دنباله ای از انتخابها با شروع از آدرس پایه ساختمان اصلی و محاسبه یک آفست برای یافتن محل عنصر اولین سطح و سپس محاسبه یک آفست از این آدرس پایه برای یافتن عناصر دومین سطح و غیره است.

ساختمان داده ها (ادامه)

```
struct date {
   int d;
   int m;
   int y;
};
struct st {
   int no;
   float ave;
   date enter;
   char zip[3];
};
st s[5];
```

ساختمان داده ها (ادامه)



ساختمان داده ها (ادامه)

لیست ها

- مشخصات و نحو: لیستها همانند بردارها حاوی دنباله مرتبی از اشیا هستند. لیست ها معمولا دارای طول متغیر می‌باشند.
- اولین عنصر لیست را معمولاً راس می گویند.

ساختمان داده ها (ادامه)

لیست ها(ادامه)

- پیاده سازی: مدیریت حافظه منظم که برای بردارها و آرایه ها مفید است در اینجا قابل استفاده نیست.
- معمولاً از سازمان مدیریت حافظه پیوندی استفاده می شود.
- لیست سه فلید اطلاعات نیازدارد:
  - یک فیلد نوع
  - دو فیلد اشاره گر لیست

ساختمان داده ها (ادامه)

لیست ها(ادامه)

- شکلهای گوناگون لیستها:
- پشته ها و صفها
- درختها
- گرافهای جهت دار
- لیستهای خاصیت(همان رکورد متغیر)

ساختمان داده ها (ادامه)

مجموعه ها

مجموعه شی داده ای است که شامل مقادیر نامرتب و مجزا است.

عملیات اصلی روی مجموعه ها عبارتند از:

عضویت

درج و حذف یک مقدار

اجتماع و اشتراک

ساختمان داده ها (ادامه)

مجموعه‌ها به طور کلی به دو دسته تقسیم می‌شوند:

مجموعه متناهی

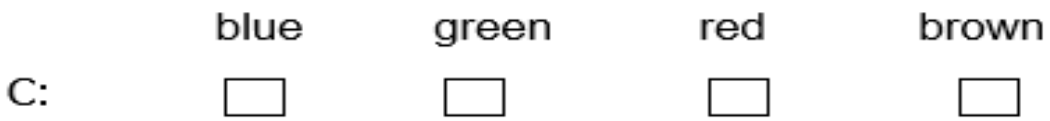
مجموعه نامتناهی

دستورات زیر را در پاسکال ببینید:

```
type
    s  set of integer;
    c  set of(blue, green, red, brown);
var
    s1,s2 : s;
    c1,c2 : c;
```

ساختمان داده ها (ادامه)

پیاده سازی: برای پیاده سازی مجموعه های متناهی از نمایش بیتی و مجموعه های نامتناهی از درهم سازی استفاده می شود. به عنوان مثال، برای پیاده سازی مجموعه cمی توان از ۴ بیت استفاده کرد که هر بیت نشان دهنده یک عضو مجموعه است:



c1 = [green, brown] ⇒ c1 : 0 1 0 1

c1 = c1 + [red] ⇒ c1 : 0 1 1 1

ساختمان داده ها (ادامه)

براي پياده سازي مجموعه s1، جدولي مثلاً با 10سطر و دو ستون در نظر مي گيريم تا عناصر مجموعه در آن ذخيره شوند.

يك تابع درهم سازي به صورت زير در نظر مي گيريم:

hash(x) = (x mod 10) + 1

مقداري كه توسط تابع hash برگردانده مي شود، سطري از جدول را مشخص مي كند كه x بايد در آن جا قرار گيرد:

s1 = [];
s1 = [34];
hash(34) = (34 mod 10) + 1 = 5

ساختمان داده ها (ادامه)

عدد 35 باید در سطر 5 قرار گیرد:

s1 = s1 + [44];

hash(44) = (44 mod 10) + 1 = 5

در سطر 5 قبلاً عدد 34 قرار گرفته است. این حالت را برخورد گویند. براي رفع برخورد به سه روش مي توان عمل کرد:

1. درهم سازي مجدد
2. باکت بندي
3. پيمايش ترتيبي

انواع داده انتزاعی

تکامل مفهوم نوع داده

مفهوم اولیه نوع داده نوع را به صورت مجموعه ای از مقادیر تعریف می کند که یک متغیر می تواند آنها را بپذیرد.

نمایش حافظه مربوط به مقادیر حقیقی و صحیح کاملا بسته بندی شده است یعنی از برنامه نویس پنهان است.

برنامه نویس بدون اینکه از جزئیات نمایش حافظه این انواع اطلاع داشته باشد از اشیای داده آنها استفاده می کند.

برنامه نویس فقط نام نوع و عملیاتی را برای دستکاری آن نوع فراهم می بیند

انواع داده انتزاعی

تکامل مفهوم نوع داده

انتزاع داده ها

نوع داده انتزاعی :

- مجموعه ای از اشیای داده
- مجموعه ای از عملیات انتزاعی بر روی آن انواع داده
- بسته بندی تمام آنها به طوری که کاربر نوع جدید نتواند اشیای داده ای از آن نوع را به جز از طریق عملیاتی که برای آن تعریف شده است دستکاری کند.

انواع داده انتزاعی

پنهان سازی اطلاعات

برای نوشتن برنامه بزرگ باید از استراتژی تقسیم و حل استفاده کرد.

طراحی ماژول معمولاً به دوروش انجام می شود:

ماژولهای تجزیه تابعی

ماژولهای تجزیه داده ای

انواع داده انتزاعی

پنهان سازی اطلاعات(ادامه)

زبان برنامه سازی انتزاع را به دو روش پشتیبانی می کند:

- با تدارک کامپیوتر مجازی که کاربرد آن ساده تر و قدرت آن بیش از کامپیوتر سخت افزار است.
- زبان امکاناتی را فراهم می کند که برنامه نویس می تواند انتزاعها را به وجود آورد.

زیربرنامه ها مکانیزم بسته بندی را شکل می دهند که تقریباً در هر زبانی وجود دارد.

بسته‌بندی با زیر برنامه‌ها

دو دیدگاه از زیربرنامه در اینجا مهم است:

- سطح طراحی برنامه: زیر برنامه‌ها عمل مجرد تعریف شده توسط برنامه‌نویس را نشان می‌دهند.
- سطح طراحی زبان: توجه بر روی طراحی و پیاده‌سازی امکانات عمومی برای تعریف فراخوانی زیر برنامه‌ها می‌باشد.

بسته‌بندی با زیر برنامه‌ها

- ✓ زیر برنامه ها و عملیات انتزاعی
- مشخصات زیربرنامه:
  - نام
  - امضای زیربرنامه :تعداد پارامترها، ترتیب آن ها، نوع هر کدام، تعداد نتایج و نوع آن ها را مشخص می کند.
- فعالیتی که توسط زیربرنامه انجام می شود.
- ✓ پیاده سازی زیربرنامه شامل:
- پیاده سازی توسط بدنه زیربرنامه تعریف می شود که متشکل از اعلان داده های محلی است.
- دستوراتی که عملکرد زیربرنامه را مشخص می کند.

بسته‌بندی با زیر برنامه‌ها

تعریف و فراخوانی زیربرنامه

تعریف زیربرنامه خاصیت ایستای یک برنامه است.

درحین اجرای برنامه اگر زیربرنامه ای فراخوانی شود سابقه فعالیتی از آن زیربرنامه ایجاد می شود.

تعریف زیربرنامه قالبی برای ایجاد سابقه فعالیت در حین اجرا است.

شی داده در حین اجرا برنامه ایجاد می شود:

- در حین ورود به زیربرنامه
- توسط عملیاتی مثل malloc

بسته‌بندی با زیر برنامه‌ها

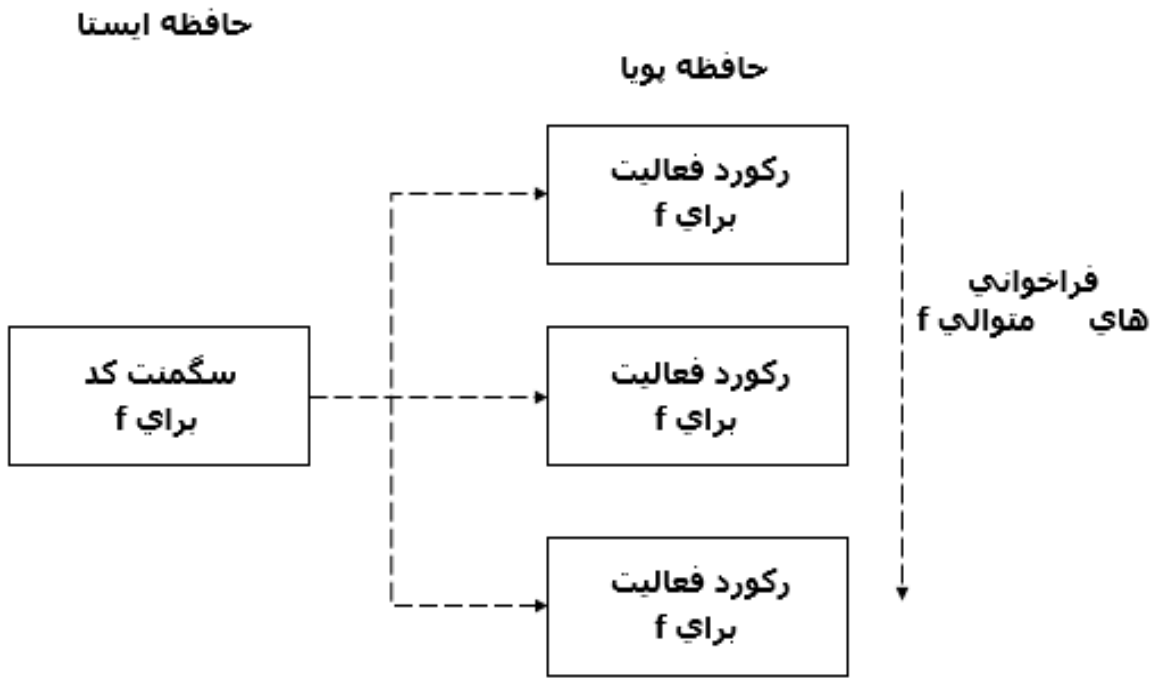
تعریف و فراخوانی زیربرنامه(ادامه)

پیاده سازی تعریف و فراخوانی زیربرنامه

الگو به دو بخش تقسیم می شود:

- بخش ایستا که سگمنت کد نام دارد و حاوی ثوابت و کد اجرایی است.
- بخش پویا که رکورد فعالیت نام دارد.

بسته‌بندی با زیر برنامه‌ها



بسته‌بندی با زیر برنامه‌ها

تعریف زیربرنامه به عنوان اشیای داده

ترجمه عملیاتی است که تعریف زیربرنامه را به شکل رشته کاراکتری گرفته شی داده زمان اجرا را تولید می کند که این تعریف را نمایش می دهد.

اجرا عملیاتی است که تعریفی به شکل زمان اجرا را گرفته سابقه فعالیتی را از آن ایجاد می کند و آن سابقه فعالیت را اجرا می نماید.

تعریف نوع

در تعریف نوع، نام تعیین می‌شود و اعلانی وجود دارد که ساختار دسته‌ای از اشیای کلاس را توصیف می‌کند به این ترتیب نام نوع به عنوان نام دسته‌ای از اشیا داده‌ای محسوب می‌شود هر وقت به اشیای داده‌ای از آن ساختار نیاز باشد به جای تکرار توصیف ساختمان داده کافی است نام نوع ارایه شود.

```
Strict rational{
int num1;
Int num2;
};
Struct rational a,b,c;
```

تعریف نوع

کنترل نوع چه به صورت ایستا و چه به صورت پویا مقایسه‌ای بین نوع آرگومان واقعی و نوع آرگومانی است که برای آن عمل مورد انتظار است.

دو راه برای مشخص کردن تساوی نوع وجود دارد:

هم‌ارزی نام: دو نوع داده زمانی هم ارزند که نام یکسانی داشته باشند.

هم‌ارزی ساختاری: اشیای داده آن ها ساختمان داده یکسانی داشته باشند.

معایب هم ارزی نوع:

- هر شی که در انتساب بکار می رود باید دارای نام باشد.
- یک تعریف نوع باید در سراسر برنامه یا بخش بزرگی از برنامه قابل استفاده باشد.

تعریف نوع

هم ارزی نوع(ادامه)

هم ارزی ساختاری

معایب

- آیا ترتیب فیلدها باید یکی باشد ...
- دو متغیر ممکن است به طور تصادفی از نظر ساختاری یکسان باشند.
- تعیین هم ارزی ساختاری در مورد انواع پیچیده هزینه ترجمه دارد.
  - تساوی اشیای داده
  - تساوی پشته
  - تساوی مجموعه