Data structures

31 December 2021 12:14

Data Structure

- Arrays, matrices, Linked List
- Stack, Queues, Trees, graph, Hashing
- Recursion, Sorting

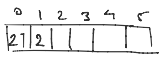
level 1
How they work
level 2
internal detail

level 3
How to code

Arrays

Intro

```
int A[5];
A[0] = 27;
A[1] = 2
```



eg.

```
int main(){
  int A[5]
  int B[5] = {2,3,4,8,9};
  for(int i=0 ; i<5 ; i++){
      cout << B[i] <<endl;
  }
}
```

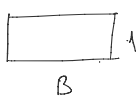
o/p
2
3
4
8
9

Structure

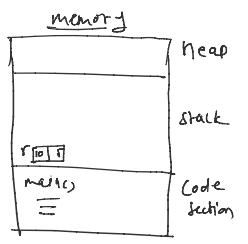
Collection of Data members, related datamember under one name

→ Similar type / Dissimilar type

used to define user define datatype




```
Struct Rect{
  int length;
  int Bread;
};
main(){
  Struct Rect r;   → declaration
  Struct Rect r={10,5};  → initial+declr
  r.length = 5;
  r.Breadth = 20;
}
```



eg. Complex number

a+ib

```
Struct Complex{
  int real;
  int img;
};
```

eg. Cards

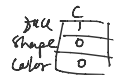
face = 1,2,3, ---- 10, J, Q, K (11, 12, 13)

shape = 0 1 3 4

Color = 0 (Black), 1 (Red)

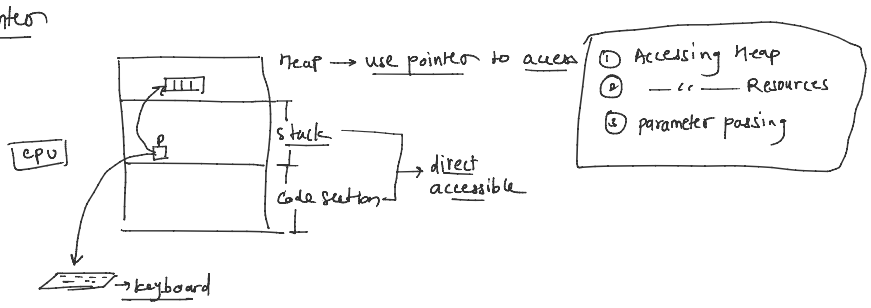
playing card

```
Struct card {
  int face;
  int Shape;
  int Color;
};
int main(){
  struct card c;
  c.face = 1;
  c.Shape = 0;
  c.Color = 0;
  Struct card c = {1,0,0};
}
```

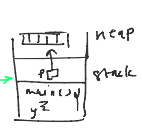



```
int main(){
  Struct Card deck[52] =
  {{1,0,0},{2,0,0} ----
   {1,1,0},{2,1,0} ---- };
}
```

Pointer

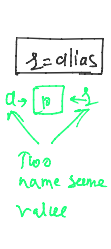



```
main(){
  int *p;
  P = new int[5];
  p[0] = 10; P[1] = 2; P[2] = 20; P[3] = 14; P[4] = 9;
  for(int i = 0; i<5; i++){
      Cout << P[i] <<endl;
  }
  delete [] p;   → delete allocated memory
  return 0;      ☆ It prevent memory leak
}
```



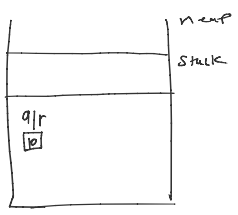
o/p = 10, 2, 20, 14, 9

Reference




```
int main(){
  int a = 10;
  int &1 = a;
  1++;
  Cout <<1;
}
```

o/p → 11



Pointer to Structure

```
Struct Rect{
  int length;
  int breadth;
};
int main(){
  Struct Rect r={10,5};
  Struct Rect *p = &r;
  r.length = 15;   → Normal variable
  (*p).length = 20;
  p → length = 22;   → Pointer
}
```