Structures and Unions

Data and File Structures Laboratory

http://www.isical.ac.in/~dfslab/2017/index.html

Structures

Definition

A structure is a collection of one or more variables, possibly of different types, grouped together under a single name for convenient handling.

Example:

```
1  struct point {
2      float x; // x and y are called
3      float y; // members or fields
4  } p1, p2;
5
6  struct triangle {
7      struct point a, b, c;
8  } t;
```

Operations on structures

- Assignment to members / fields

```
p1.x = 1.0; p1.y = 2.0; t.a.x = 0.0; t.a.y = 0.5;
```

- Assignment / copying of structure variables

```
struct triangle t1, t2;   ...   ; t2 = t1;
```

- Structures may be passed to functions, and returned by functions.
- **But comparison operators (==, !=) don't work!**

Structure operations

- Initialisation

```
1  struct point {
2      float x; // x and y are called
3      float y; // members or fields
4  } p1, p2;
5
6  struct triangle {
7      struct point a, b, c;
8  } t = { { 1.0, 1.0 },
9          { -1.0, 1.0 },
10         { 1.0, -1.0 } };
```

Typedefs

```
1  typedef unsigned int Length;
2  Length len, maxlen;
3  Length lengths[];
4
5  typedef char *String;
6  String p, myStrings[128]; // p - single string,
       myStrings - array of strings
7  int strcmp(String, String);
8  p = (String) malloc(100);
```

Typedefs

```
1  typedef struct {
2      float x;
3      float y;
4  } POINT;
5  POINT p1, p2;
6
7  typedef struct {
8      struct point a, b, c;
9  } TRIANGLE;
10 TRIANGLE t;
```

Pointers to structures

```
1  struct point *pp; // old scheme (before typedef)
2
3  POINT *pp; // new scheme (after typedef)
4
5  (*pp).x = (*pp).y = 0.0; // OR
6  pp->x = pp->y = 0.0;
```

Memory allocation

```
1  TRIANGLE *tp;
2
3  tp = (TRIANGLE *) malloc(num_triangles *
       sizeof(TRIANGLE));
```

Acknowledgements

- `http://cse.iitkgp.ac.in/~pds/notes/`
  (please see the above page for many more practice problems)