2a-struct

```
struct Flight {
     int id;
     char path[10];
};

struct Captain {
    char sign[4];
    char base[4];
};

struct Flight flight;

int main(int argc, char**argv)
{
    struct Flight *pF1 = (struct Flight *)malloc(sizeof(struct Flight));

    struct Flight *pF2 = (struct Flight *)malloc(sizeof(struct Flight));

    struct Captain *pC1 = (struct Flight *)malloc(sizeof(struct Flight));

    return 0;

}
```

Heap Memory

id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
sign[0]	sign[1]	sign[2]	sign[3]	base[0]	base[1]	base[2]	base[3]					

```
struct flight_structure {
     int id;
     char path[10];
} Flight;

struct captain_structure {
    char sign[4];
    char base[4];
} Captain;

Flight flight;

int main(int argc, char**argv)
{
    Flight *pF1 = (Flight *)malloc(sizeof(Flight));

    Flight *pF2 = (Flight *)malloc(sizeof(Flight));

    Captain *pC1 = (Flight *)malloc(sizeof(Flight));

    return 0;

}
```

Heap Memory

id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
id	path[0]	path[1]	path[2]	path[3]	path[4]	path[5]	path[6]	path[7]	path[8]	path[9]		
sign[0]	sign[1]	sign[2]	sign[3]	base[0]	base[1]	base[2]	base[3]					

Instead of always writing **struct this variable** ....
One can declare a struct as a type.

This is done by using the reserved word **typedef**.

```
typedef struct abc ABC;
```

Then you can write

```
ABC myAbcVariable;
```