`struct` keyword is used to define a data structure

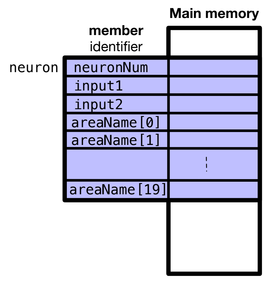
```
struct Neuron {
  int neuronNum;
  double input1, input2;
  char areaName[20];
};
```

members or fields of `struct Neuron`

```
struct Neuron {
  int neuronNum;
  double input1, input2;
  char areaName[20];
} neuron;
```

define a data structure

declare a variable of type `struct Neuron`




**Define and redefine `struct Neuron` separately**

```
#include <stdio.h>

struct Neuron {
  int neuronNum;
  double input1, input2;
  char areaName[20];
};

typedef struct Neuron Neuron;

int main(void) {
  Neuron neuron;
  neuron.input1 = 7.9;
  return 0;
}
```

1 Define a data structure

2 Refine `struct Neuron` to Neuron

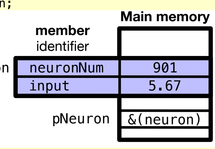
**Define and redefine `struct Neuron` in the same statement**

```
#include <stdio.h>

typedef struct Neuron {
  int neuronNum;
  double input1, input2;
  char areaName[20];
} Neuron;

int main(void) {
  Neuron neuron;
  neuron.input1 = 7.9;
  return 0;
}
```

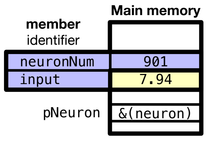
2 Refine `struct Neuron` to Neuron

1 Define a data structure

```
#include <stdio.h>

typedef struct Neuron {
  int neuronNum;
  double input;
} Neuron;

int main(void) {
  Neuron neuron = {901, 5.67};
  Neuron *pNeuron = &neuron;
```

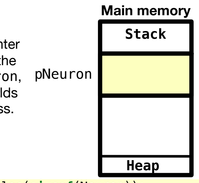



```
  (*pNeuron).input = 7.94;
```




```
  printf("neuron.input = %.2lf\n", neuron.input);
  printf("(*pNeuron).input = %.2lf\n", (*pNeuron).input);
  return 0;
}
```

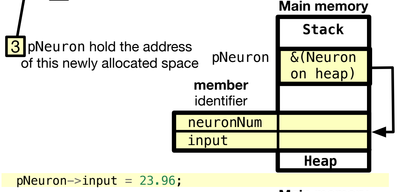
```
#include <stdio.h>
#include <stdlib.h>

typedef struct Neuron {
  int neuronNum;
  double input;
} Neuron;

int main(void) {
  Neuron *pNeuron;
```




1 `pNeuron` is a pointer that should hold the address of a `Neuron`, but it currently holds a garbage address.

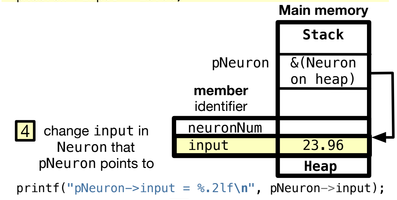
```
  pNeuron = (Neuron *)malloc(sizeof(Neuron));
```

2 allocates `sizeof(Neuron)` bytes on the heap




3 `pNeuron` hold the address of this newly allocated space

```
  pNeuron->input = 23.96;
```




4 change `input` in `Neuron` that `pNeuron` points to

```
  printf("pNeuron->input = %.2lf\n", pNeuron->input);
  free(pNeuron);
  return 0;
}
```

5 need to free memory space allocated