Unions

November 30, 2021

1 Unions

1.1 Topics

- Union type definition
- Defining and using union type
- Demonstration of how union type works

1.2 Union

- a union type is burrowed from C
- it's a user-defined type that allows you to store different data types in the same memory location
- union is usually defined with many members similar to structure, but only one member will be active and valid at a time
- it's an efficient way of using the same memory location for multiple purpose or different data types
  - only primitive types (C-types, char, int, float, double, etc.) are allowed
  - user-defined types such as string is not allowed
- see CPP reference for more: https://en.cppreference.com/w/cpp/language/union
- syntax to define **union** type:

```
union UnionName {
    type1 memberName1;
    type2 memberName2;
    ...
    typeN memberNameN;
};
```

- defining and using union type is syntatcially similar to `struct` type
- member is accessed using `.` member access operator

```
UnionName object;
object.memberName = [value];
```

1.2.1 Visualize it in Pythontutor.com

[1]:
```
#include <iostream>
#include <cstring>

using namespace std;
```

[2]:
```
// Define Union type
union Data {
  char ch;
  int ID;
  float price;
  char fullName[100];
};
```

[3]:
```
// declare objects
union Data data;
```

[4]:
```
sizeof(data)
// size of data object is the size of the largest mememberr variable
```

[4]: 100

[5]:
```
// only one member is active at a time
data.ch = 'A'
```

[5]: 'A'

[6]:
```
// ch member is active
cout << data.ch << endl;
```

```
A
```

[7]:
```
data.ID = 100;
```

[8]:
```
// ID member is active
cout << data.ID << endl;
```

```
100
```

[9]:
```
// ch is now not valid
cout << data.ch;
```

```
d
```

[10]:
```
// price member is active
data.price = 9.99;
```

[11]:
```
cout << data.price;
```

```
9.99
```

[12]:
```
// both ch and ID not active and valid
cout << data.ch << " " << data.ID;
```

```
 1092605706
```

[13]:
```
// data.fullName is now active
strncpy(data.fullName, "John Smith", 10);
```

[14]:
```
cout << data.fullName;
```

```
John Smith
```

[15]:
```
// all other 3 members are not active and valid
cout << data.ch << " " << data.ID << " " << data.price;
```

```
J 1852337994 1.79838e+28
```

1.3 Applications

- union has limited and very specific application in real world

1.4 Kattis problems

- union is not a strict requirement to solve Kattis problems

1.5 Summary

- learnd what union types are
- learned how to define and use union types

[ ]:
```

```