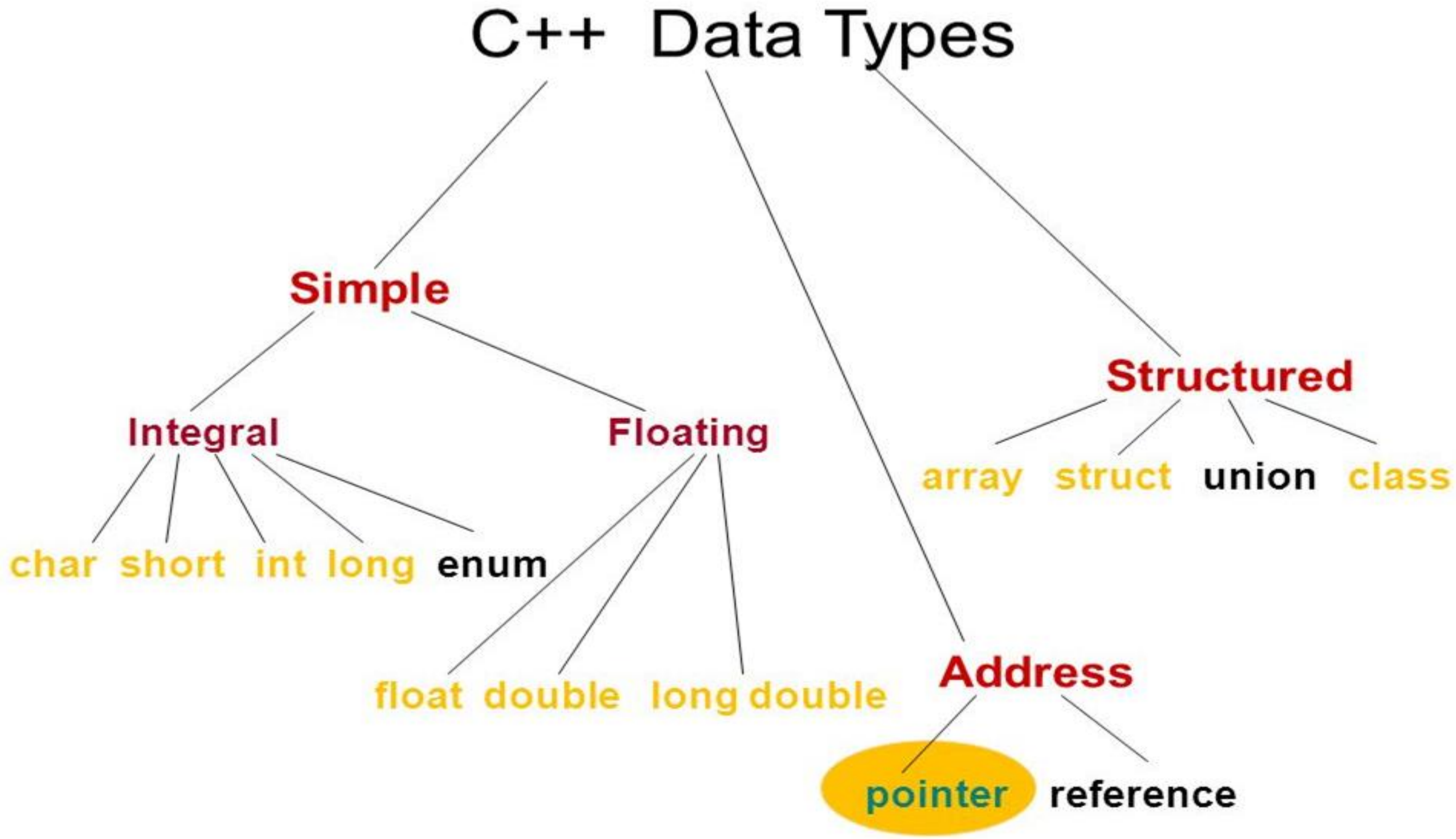
СТРУКТУРИ ДАНИХ

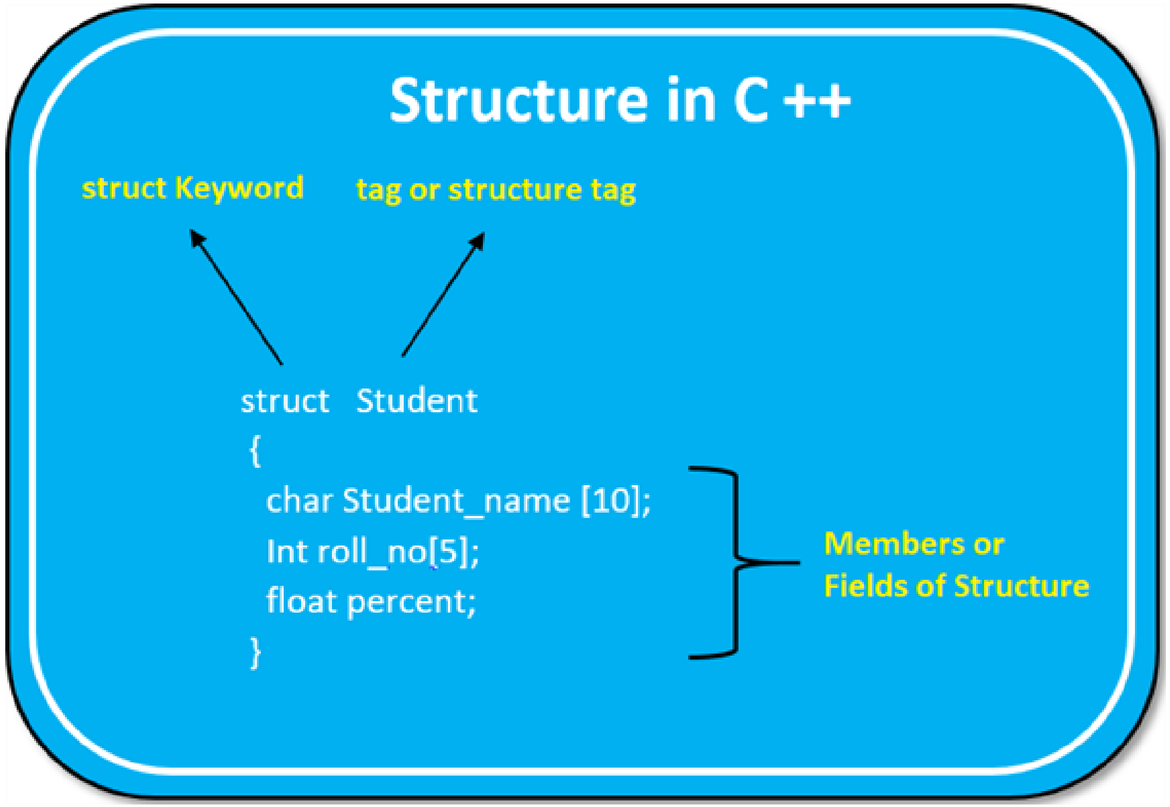
C++


C++ Data Types
Simple
Integral
Floating
char
short
int
long
enum
float
double
long double
Structured
array
struct
union
class
Address
pointer
reference

Структури

Структура - це сукупність змінних різних типів даних під єдиною назвою.

```
struct Person
{
     char name[50];
     int age;
    };
```




```
int main()
{
     Person p1;
     cin.get(p1.name, 50);
     cin>>p1.age;
}
```

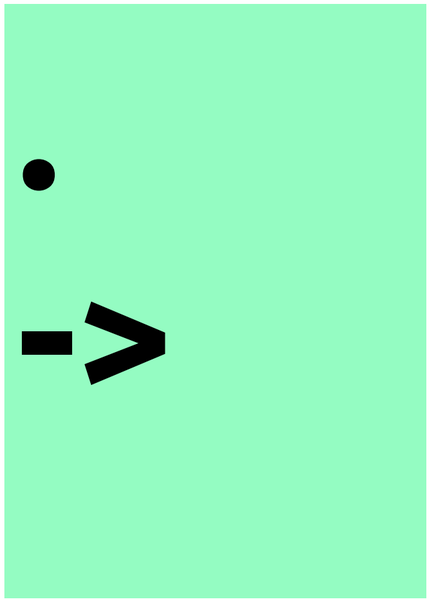
Доступ до елементів структур

Доступ до елементів структур може здійснювати двомісні операції: операцією точки (.) або операцією стрілки (->).

Якщо доступ здійснюється через об'єкт, то використовується точка
cout<<p.age;

Якщо доступ здійснюється за допомогою вказівника на об'єкті, використовується операційна стрілка:

```
p->age = 32;
```



Приклад

```
#include <iostream>
using namespace std;
struct Person
{
    char name[50];
    int age;
   };

int main()
{
    Person p1;
    cin.get(p1.name, 50);
    cin >> p1.age;
    cout << "Name: " << p1.name << endl;
    cout <<"Age: " << p1.age << endl;
    return 0;
}
```

Масив структур

```
#include <iostream>
using namespace std;
struct Person
{
    char name[50];
    int age;
   };

int main()
{
    Person p1[3];
    for(int i=0;i<3;i++)
    {
    cin.get(p1[i].name, 50);
    cin >> p1[i].age;
    cin.get();
    }
 for(int i=0;i<3;i++)

{cout << "Name: " << p1[i].name << endl;
    cout <<"Age: " << p1[i].age << endl;
}
    return 0;
}
```

Структури і функції

Перевагою використання структур є можливість передати всю структуру в функцію, яка повинна працювати з її елементами

```
#include <iostream>
 struct Employee
{short id;
    int age;
    double salary;
};
void printInformation(Employee employee)
{
    std::cout << "ID: " << employee.id;
    std::cout << "Age: " << employee.age;
    std::cout << "Salary: " << employee.salary;
}
 int main()
{    Employee john = { 21, 27, 28.45 };
    Employee james = { 22, 29, 19.29 };
     printInformation(john);
     printInformation(james);
     return 0;
}
```

Вказівники на структури

Як і на будь-який інший тип, на структури можна вказати за власним типом вказівників:

```
struct movies_t {
    заголовок рядка;
    int рік;
};
movies_t amovie;
movies_t * pmovie;
```

Вказівники на структури

Використовуючи вказівник на структуру, можна отримати доступ до її елементів.

Два способи

застосування опису операцій пошуку:

- **(*вказівник _на_структуру).ім’я_елемента**

використання операцій -> (операція стрілка):

- **вказівник _на_структуру->ім’я_елемента**

```
struct person kate = {31, "Kate"};
    struct person * p_kate = &kate;

    char * name = p_kate->name;
    int age = (*p_kate).age;

    // змінимо елемент age
   p_kate->age = 32;
```

Приклад

Приклад, який поєднує вказівники та структури та служить для введення нового оператора: оператора стрілки (->):

```
string mystr;
 movies_t amovie;
 movies_t * pmovie;
 pmovie = &amovie;

  getline (cin, pmovie->title);
  getline (cin, mystr);
 (stringstream) mystr >> pmovie-
>year;
   cout << pmovie->title;
```