Advanced Programming in C++

فرشاد حکیم پور

1

Pointers Complete picture

- Reference variables are similar to pointers
- They are defined by "&" prefix
- After their definition use them just like regular variable
- You will NOT be able to chenge were they are referring to

```
int i = 15;
int &x = i;
x *= 2;
std::cout << i << " == " << x << std::endl;
```

2

Pointers Complete picture

- Prefix "&" and "*"
  - Used in definitions

```
int *iPtr = new int(5);
std::cout << *iPtr << std::endl;
. . .
int &i = *new int(5);
std::cout << i << std::endl;
```

3

Pointers Complete picture

- Unary operator "&" and "*"
  - Used in operation for retrieving address and dereferencing

```
int i;
int *iPtr;
i = 25;
iPtr = &i;
std::cout <<*iPtr<< std::cout;
```

- Binary operator "&" and "*"
  - Used for multiplication and bitwise logical *and*

4

Function Pointers

- You can define a pointer to a function just like a pointer to an array or an object
- A pointer to a function is just the address of where the execution code for the function starts

5

Example

```
class MyArrayProcessor
{
    private:
        vector<int> array;
    public:
    MyArrayProcessor(vector<int> _a)
    {
        array.assign(_a.begin(),_a.end());
    }

    void printAggregate(double (*aggr)(vector<int> a))
    {
        std::cout << (*aggr)(array) << std::endl;
    }
};
```

6

Example

```
double sum(vector<int> a)   {
    double sum = 0;
    for (int i = 0 ; i<a.size(); i++)
       sum += a[i];
    return sum;             }

double avg(vector<int> a) {return (sum(a) / a.size());}

double max(vector<int> a)   {
    double max = std::numeric_limits<int>::min();
    for (int i = 0 ; i<a.size(); i++)
       if (a[i] > max ) max = a[i];
    return max;             }

double min(vector<int> a)   {
    double min = std::numeric_limits<int>::max();
    for (int i = 0 ; i<a.size(); i++)
       if (a[i] < min ) min = a[i];
    return min;             }
```


Example

```
int main(int argc, char *argv[])
{
    int size = 6;
    vector<int> array;
    for (int idx = 0; idx<size; idx++)
        array.push_back(idx*2);
    MyArrayProcessor ap = MyArrayProcessor(array);
    ap.printAggregate(sum);
}
```


Exercise 5

- Write a program that reads an image and writes a new one after applying a filter function.
- The filter function calculates a new value for each pixel depending on the neighboring pixels (e.g. a weighted average of nine pixels)
- Pass the filter function by pointer to your image processing class
- Deadline Wednesday 11th Day
- Send code to tamrin.ut@gmail.com