Advanced Programming in C++

فرشاد حکیم پور

Constant (Read-only) Data

- "const" qualifier is used to inform the compiler that the value of a variable must not change

```
const double pi = 3.14;
pi = 3.1415;   // Error!
```


Constant (Read-only) Data

- "const" qualifier can be used with pointers in two different ways (or their combination)
  1. The value at the location where the pointer is pointing can not be changed *through the pointer*
  2. The pointer is fixed and cannot change to point to a different location

Constant (Read-only) Data

- The value at the location where the pointer is pointing can not be changed *through the pointer*

```
double pi = 3.14;
const double *piPtr = new double(3);
piPtr = &pi;
cout << *piPtr << endl;  // 3.14
*piPtr = 3.1415;         // Error!
pi = 3.1415              // Valid
cout << *piPtr << endl;  // 3.1415
```


Constant (Read-only) Data

- The pointer is fixed and cannot change to point to a different location

```
double pi = 3.14;
double *const piPtr = &pi;
piPtr = new double (3.1415);  // Error!
*piPtr = 3.1415;              // Valid
cout << pi << endl            // 3.1415
```


"const" in method declarations

- Passing pointers to constant values
  - Pointer can change to different locations but the value it points to may not change

```
void addressParser(const char *addrPtr)
{
  for (; *addrPtr != '\0'; addrPtr++)
  {
     *addrPtr = 'x'; // Error!
     . . .
  }
}
```

"const" in method declarations

- Passing pointers to constant values
  - Pointer can change to different locations but the value it points to may not change

```
void addressParser(char *const addrPtr)
{
  for (; *addrPtr != '\0'; addrPtr++) // Error!
  {
     *addrPtr = 'x';
     . . .
  }
}
```


Throwing an Exception

- Your programs can throw exception as well

```
#include <stdexcept>

double divide(double a, double b)
{
    if (b == 0) throw runtime_error("devide by zero")
    return (a/b);
}
```


Throwing Exceptions

- If your program discovers an error that cannot handle it (often due to problem with the input data), it should throw an exception
- The caller program is to handle the error
  - A program may also re-throw an exception to its own caller

How to Throw an exception

- If an error occurred (that your program cannot handle)
  1. Create an exception object
  2. Throw the exception object

Creating an Exception Object

```
class DivideByZero : public runtime_error
{
public:
  DivideByZero() :
    runtime_error("Devision by zero"){}
};
```


Throwing an Object

```
double divide(double dividend, double divisor)
{
   if (divisor == 0)
      throw DivideByZero();
   double quotient = dividend / divisor;
   return quotient;
}
```

Catching the Exception

```
try
   {
       double a = divide(5, 0);
       cout << a << endl;
   }
   catch(DivideByZero &e)
   {
       cout << e.what() << endl;
   }
```


Exercise Six

- Using inheritance capabilities of objects in C++ design and implement classes polygon and circle as subclasses of shape. (area, perimeter, etc.)
- Throw exceptions in case of erroneous invocations. (e.g. adding point after closing a polygon)
- Deadline Tuesday 24th day
- Send the code to tamrin.ut@gmail.com