

计算机科学基础II

实验十七 模板与类参数

曹鹏

Email: caopeng@seu.edu.cn

Tel: 13851945861

实验目的

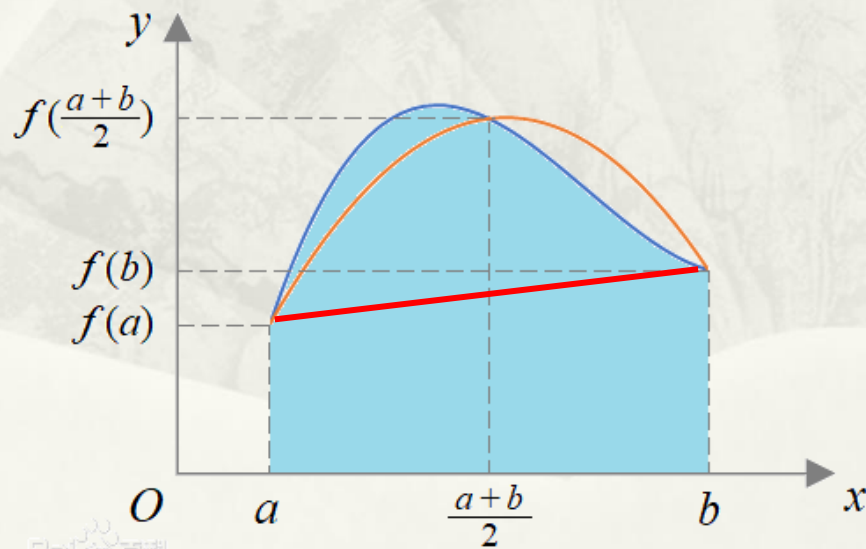
实验十七

1. 学会利用类对象作为参数，通过运算符的重载，实现模板更好的通用性。
2. 学会利用类参数传递函数，取代传统的函数指针。

实验十七 2. 编程

求函数 $y=f(x)$ 在区间 $[a,b]$ 的近似定积分

- ◆ 梯形法：用直线来近似曲边梯形的曲边
- ◆ 辛普生法：用抛物线来近似曲边梯形的曲边

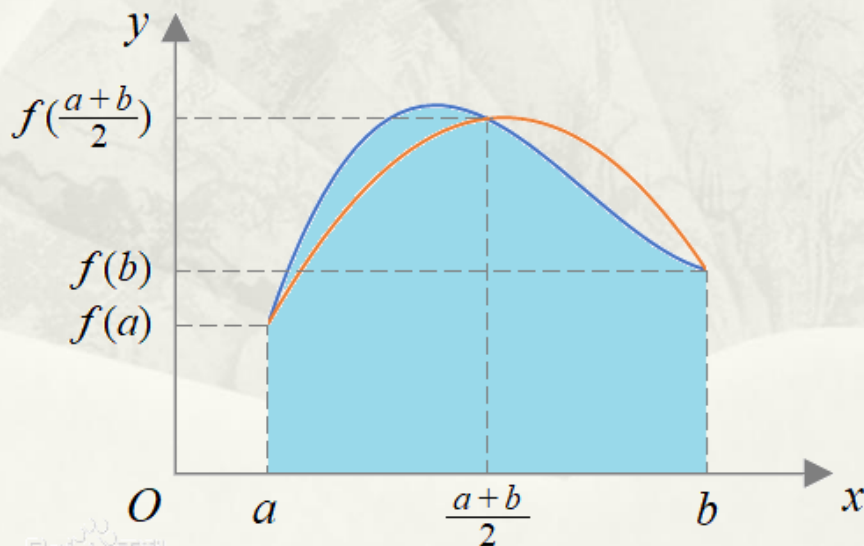


实验十七 2. 编程

辛普生法求函数 $f(x)$ 在区间 $[a,b]$ 的近似定积分

- ◆ 将区间 $[a,b]$ 等分为 n 份， n 为偶数，每份区间 Δx
- ◆ 通过以下公式求近似定积分

$$\int_a^b f(x)dx \approx \frac{1}{3} \Delta x [y_0 + y_n + 4(y_1 + y_3 + \dots + y_{n-1}) + 2(y_2 + y_4 + \dots + y_{n-2})]$$



实验十七 2. 编程

辛普生法求被积函数 $f(x)$ 在区间 $[a,b]$ 的近似定积分

◆ 将不同的被积函数 $f(x)$ 定义为不同的类

- ◆ 积分区间 $[a,b]$ 和份数 n 定义为成员数据 a,b,n
- ◆ 求自变量为 x 时函数值的方法定义为成员函数 $\text{fun}(x)$
- ◆ 求自变量为 x 时原函数值的方法定义为成员函数 $\text{prim}(x)$
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

◆ 辛普森法求不同被积函数近似定积分动作序列一致，因此定义为类模板

- ◆ 函数 $f(x)$ 的类对象定义为模板类型参数
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

被积函数类F1/F2/F3

```
class F1{
public:
    double a, b; //定积分区间[a,b]
    F1(double aa=0, double bb=0);
    double fun(double x); //计算函数值
    double prim(double x); //计算原函数值
    double integrate() { return prim(b) - prim(a); } //计算定积分值
};
```

实验十七 2. 编程

辛普生法求被积函数 $f(x)$ 在区间 $[a,b]$ 的近似定积分

◆ 将不同的被积函数 $f(x)$ 定义为不同的类

- ◆ 积分区间 $[a,b]$ 和份数 n 定义为成员数据 a,b,n
- ◆ 求自变量为 x 时函数值的方法定义为成员函数 $\text{fun}(x)$
- ◆ 求自变量为 x 时原函数值的方法定义为成员函数 $\text{prim}(x)$
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

◆ 辛普森法求不同被积函数近似定积分动作序列一致，因此定义为类模板

- ◆ 函数 $f(x)$ 的类对象定义为模板类型参数
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

类	被积函数 $\text{fun}(x)$	原函数 $\text{prim}(x)$
F1	$f_1(x) = 1 + x + 2x^2$	$\int f_1(x)dx = x + \frac{1}{2}x^2 + \frac{2}{3}x^3$
F2	$f_2(x) = 1 + x + 2x^2 + 3x^3$	$\int f_2(x)dx = x + \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{3}{4}x^4$
F3	$f_3(x) = 1 + x + 2x^2 + 3x^3 + 4x^4$	$\int f_3(x)dx = x + \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{3}{4}x^4 + \frac{4}{5}x^5$

实验十七 2. 编程

辛普生法求被积函数 $f(x)$ 在区间 $[a,b]$ 的近似定积分

◆ 将不同的被积函数 $f(x)$ 定义为不同的类

- ◆ 积分区间 $[a,b]$ 和份数 n 定义为成员数据 a,b,n
- ◆ 求自变量为 x 时函数值的方法定义为成员函数 $\text{fun}(x)$
- ◆ 求自变量为 x 时原函数值的方法定义为成员函数 $\text{prim}(x)$
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

◆ 辛普森法求不同被积函数近似定积分动作序列一致，因此定义为类模板

- ◆ 函数 $f(x)$ 的类对象定义为模板类型参数
- ◆ 求函数定积分值的方法定义为成员函数 $\text{Integrate}()$

```
template<typename T>class Integer{  
    T cf;//被积函数类对象  
public:  
    Integer(T f){ cf=f; }  
    double integrate();  
};
```


实验十七 2. 编程

```
int main(){  
    F1 f1(0.0,3.0);  
  
    Integer<F1> integer1(f1);  
    cout << "函数F1在区间[0.0, 3.0]定积分" << endl;  
    cout << "精确值: " << f1.integrate() << endl;  
    cout << "辛普森法(n=1 ): " << integer1.integrate(1) << endl;  
    cout << "辛普森法(n=5 ): " << integer1.integrate(5) << endl;  
    cout << "辛普森法(n=10 ): " << integer1.integrate(10) << endl;  
    cout << "辛普森法(n=100): " << integer1.integrate(100) << endl;
```

函数F1在区间[0.0, 3.0]定积分

精确值: 25.5

辛普森法(n=1): 23

辛普森法(n=5): 21.88

辛普森法(n=10): 25.5

辛普森法(n=100): 25.5

实验十七 2. 编程

```
F2 f2(0.0,3.0);
Integer<F2> integer2(f2);
cout << "函数F2在区间[0.0, 3.0]定积分" << endl;
cout << "精确值: " << f2.integrate() << endl;
cout << "辛普森法(n=1 ): " << integer2.integrate(1) << endl;
cout << "辛普森法(n=5 ): " << integer2.integrate(5) << endl;
cout << "辛普森法(n=10 ): " << integer2.integrate(10) << endl;
cout << "辛普森法(n=100) " << integer2.integrate(100) << endl;
F3 f3(0.0,3.0);
Integer<F3> integer3(f3);
cout << "函数F3在区间[0.0, 3.0]定积分" << endl;
cout << "精确值: " << f3.integrate() << endl;
cout << "辛普森法(n=1 ): " << integer3.integrate(1) << endl;
cout << "辛普森法(n=5 ): " << integer3.integrate(5) << endl;
cout << "辛普森法(n=10 ): " << integer3.integrate(10) << endl;
cout << "辛普森法(n=100) " << integer3.integrate(100) << endl;
return 0;
}
```

函数F2在区间[0.0, 3.0]定积分

精确值: 86.25

辛普森法(n=1): 104

辛普森法(n=5): 71.2576

辛普森法(n=10): 86.25

辛普森法(n=100): 86.25

函数F3在区间[0.0, 3.0]定积分

精确值: 280.65

辛普森法(n=1): 428

辛普森法(n=5): 226.467

辛普森法(n=10): 280.663

辛普森法(n=100): 280.65



End