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

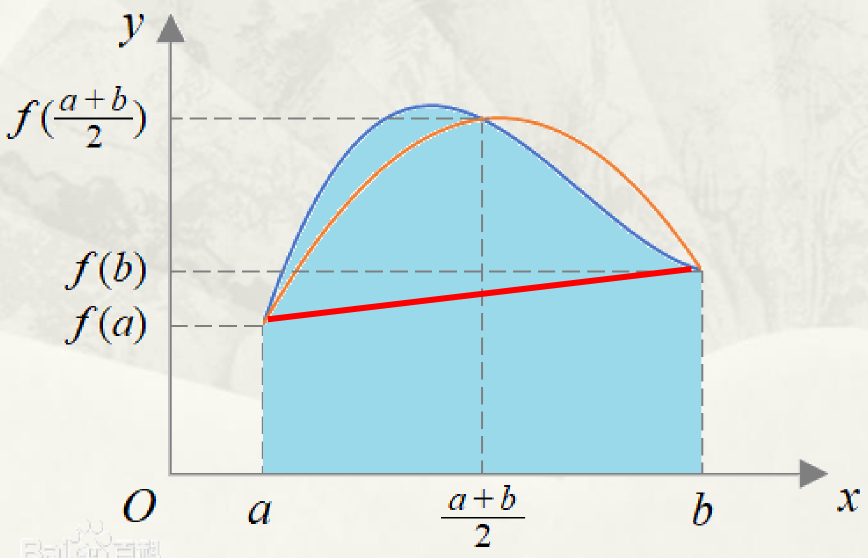
实验十七

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

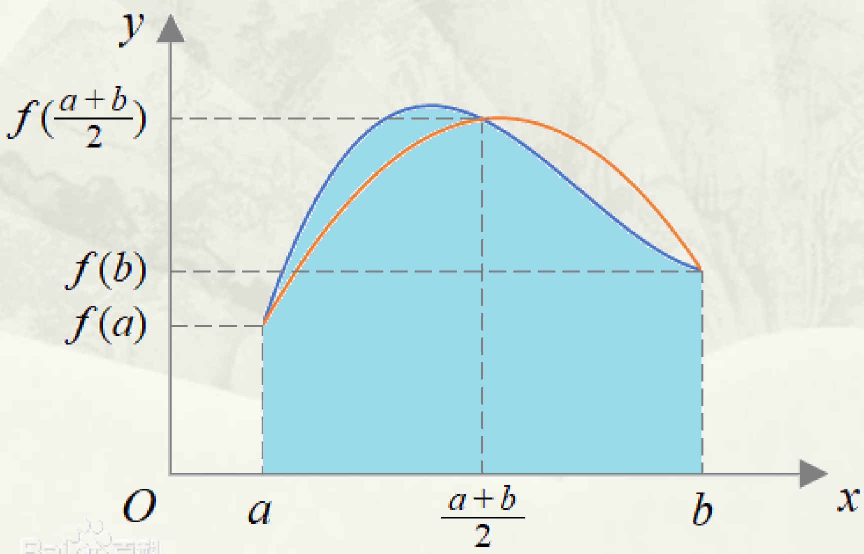


实验十七 2. 编程

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，每份区间Δx
- ◆通过以下公式求近似定积分

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



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fun(x)
  - 求自变量为x时原函数值的方法定义为成员函数prim(x)
  - 求函数定积分值的方法定义为成员函数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fun(x)
  - 求自变量为x时原函数值的方法定义为成员函数prim(x)
  - 求函数定积分值的方法定义为成员函数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Integrate()

类	被积函数 fun(x)	原函数 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fun(x)
  - 求自变量为x时原函数值的方法定义为成员函数prim(x)
  - 求函数定积分值的方法定义为成员函数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Integrate()

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

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  ):
    cout << "辛普森法(n=10 ):
    cout << "辛普森法(n=100)
    F3 f3(0.0,3.0);
    Integer<F3> integer3(f3);
    cout << "函数F3在区间[0.0,
    cout << "精确值： " << f3.i
    cout << "辛普森法(n=1  ):
    cout << "辛普森法(n=5  ):
    cout << "辛普森法(n=10 ):
    cout << "辛普森法(n=100)
    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
```

实验十七 2. 编程

被积函数类F1 $f_1(x) = 1 + x + 2x^2$

```
class F1 {
public:
        double a, b;
        F1(double aa = 0, double bb = 0) { a = aa; b = bb; }
        double fun(double x) { return (1 + x + 2 * x * x); }
        double prim(double x) { return (x + x * x / 2 + x * x * x * 2 / 3); }
        double integrate() { return prim(b) - prim(a);}
};
```

实验十七 2. 编程

被积函数类F2 $f_2(x) = 1 + x + 2x^2 + 3x^3$

```
class F2 {
public:
		double a, b;
		F1(double aa = 0, double bb = 0) { a = aa; b = bb; }
		double fun(double x) { return (1 + x + 2 * x * x + 3 * x * x
* x); }
		double prim(double x) { return (x + x * x / 2 + x * x * x * 2
/ 3 + x * x * x * x * 3 / 4); }
		double integrate() { return prim(b) - prim(a);}
};
```

实验十七 2. 编程

被积函数类F3 $f_3(x) = 1 + x + 2x^2 + 3x^3 + 4x^4$

```
class F3 {
public:
		double a, b;
		F1(double aa = 0, double bb = 0) { a = aa; b = bb; }
		double fun(double x) { return (1 + x + 2 * x * x + 3 * x * x * x + 4 * x * x * x * x); }
		double prim(double x) { return (x + x * x / 2 + x * x * x * 2 / 3 + x * x * x * x * 3 / 4 + x * x * x * x * x * 4 / 5); }
		double integrate() { return prim(b) - prim(a);}
};
```

实验十七 2. 编程

辛普生法求函数近似定积分

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

```
template< typename T>
double Integer<T>::integrate(int n){
        int i;
        double step, result;
        step=(cf.b-cf.a)/n;
        result=cf.fun(cf.a)+cf.fun(cf.b);
        for(i=1;i<n;i+=2) result+=4*cf.fun(cf.a+i*step);
        for(i=2;i<n;i+=2) result+=2*cf.fun(cf.a+i*step);
        result*=step/3;
        return result;
}
```

End