Laboratory Programming Exercise 06
Computer Engineering 160
Fall 2018
C. Paolini

Recall from Calculus the derivative of *u(x)* with respect to *x* is defined as the limit

$$\lim_{h\to 0}\frac{u(x+h)-u(x)}{h}$$

The ratio above is known as a *forward difference*, because the function evaluation of *u(x+h)* is performed at a value *x+h* infinitesimally greater than *x*, for some infinitesimal *h*. A *backward difference*, would then be

$$\frac{u(x)-u(x-h)}{h}$$

where the function evaluation *u(x-h)* is performed at a value *x-h* infinitesimally less than *x*. Combining the two, one can compute a *centered difference*

$$\frac{u(x+h)-u(x-h)}{2h}$$

Write a code that defines the following five functions:

```
double u(double x);
double dudx(double x);
double fdiff(double x, double h);
double bdiff(double x, double h);
double cdiff(double x, double h);
```

where function *u* implements the expression $u = x^3$ and is defined as

```
double u(double x)
{
      return(pow(x,3.0));
}
```

The exact derivative

$$\frac{du}{dx} = 3x^2$$

can then be defined as a C function,

```
double dudx(double x)
{
      return(3.0 * pow(x,2.0));
}
```

For this exercise, define the three functions

```
double fdiff(double x, double h)
double bdiff(double x, double h)
double cdiff(double x, double h)
```

that implement the forward difference, backward difference, and centered difference schemes, respectively. Note: you do not need to implement any limits; just compute the ratios in each scheme for a given value of *h* passed into each function. All functions should be in the same .c file as your main function. Implement an iterative structure that varies *h* from 10^{-1} to 10^{-8}, in powers of 10, and computes *u(x)*, *du/dx*, the forward difference, the backward difference, the centered difference, and the three approximation errors for each differencing scheme. For example, the forward difference error for a particular *h* would be

$$\text{error}_h = \left| fdiff(x,h) - dudx(x) \right|$$

or in C,

```
fabs(fdiff(x,h)-dudx(x))
```

Prompt the user for a value *x* and output all variable values in your iterative structure. For example,

```
Enter x: 2.5
h: 1.00e-01, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.9510e+01, bd: 1.8010e+01, cd: 1.8760e+01, fderr: 7.6000e-01, bderr: 7.4000e-01, cderr: 1.0000e-02
h: 1.00e-02, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8825e+01, bd: 1.8675e+01, cd: 1.8750e+01, fderr: 7.5100e-02, bderr: 7.4900e-02, cderr: 1.0000e-04
h: 1.00e-03, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8758e+01, bd: 1.8743e+01, cd: 1.8750e+01, fderr: 7.5010e-03, bderr: 7.4990e-03, cderr: 1.0000e-06
h: 1.00e-04, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8751e+01, bd: 1.8749e+01, cd: 1.8750e+01, fderr: 7.5001e-04, bderr: 7.4999e-04, cderr: 1.0046e-08
h: 1.00e-05, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8750e+01, bd: 1.8750e+01, cd: 1.8750e+01, fderr: 7.5000e-05, bderr: 7.5000e-05, cderr: 1.7834e-10
h: 1.00e-06, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8750e+01, bd: 1.8750e+01, cd: 1.8750e+01, fderr: 7.5022e-06, bderr: 7.4974e-06, cderr: 2.3988e-09
h: 1.00e-07, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8750e+01, bd: 1.8750e+01, cd: 1.8750e+01, fderr: 7.1649e-07, bderr: 7.7565e-07, cderr: 2.9576e-08
h: 1.00e-08, x: 2.50, u(x): 15.62, du/dx: 1.88e+01, fd: 1.8750e+01, bd: 1.8750e+01, cd: 1.8750e+01, fderr: 4.7339e-08, bderr: 2.2497e-07, cderr: 1.3616e-07
```

In your comment header, explain what you observe with respect to how the approximation error decreases for each value of *h* in each of the three differencing schemes.

Generate a screen capture of your Eclipse IDE workspace, showing your code and the output of your program in the terminal. Please name your Eclipse IDE project *Lastname_REDID_Lab_06*. Create a ZIP file of your project folder and submit the ZIP file through Blackboard.