CS601: Software Development for Scientific Computing

Maximum Points: 40 End-semester examination 16/11/2023, 9:30AM to 11:30AM (2 hours)

Instructions: This exam has two parts. Part I is open-book, open-notes (printed/written), Calculator allowed. No other electronic devices allowed. Part II is take-home. The submission instructions for part II are the same as in programming assignments. State your assumptions (if any) clearly.

Part I **(36 points)**:

1. Suppose that $u = u(x, y)$ satisfies Laplace's equation $u_{xx} + u_{yy} = 0$ in the square region $0 < x, y < 1$. Assuming a mesh size of $h = 1/3$ use the Jacobi iteration, with starting values $u_{ij}^0 = 0$ to perform *two iterations*. The boundary data are as given below (assume horizontal=x-axis and vertical=y-axis): **(10 points)**

0.0000	0.2500	0.7500	1.0000
0.4000	u12	u22	0.8000
0.8000	u11	u21	0.4000
0.0000	0.7500	0.2500	0.0000

a) You must show the values of $u_{ij}^1$and u_{ij}^2. b) Show the values of $u_{ij}^1$and u_{ij}^2 with Gauss-seidel iteration. c) The computation involves y-point stencil in (a) and z-point stencil in (b). What are the values of y and z?

2. Matrix A has a special structure: any element of the matrix, a_{ij}, can be expressed using elements of two vectors p, q, as: $a_{ij} = (p_j - q_i)^2$, p_j, q_i denote the j[th] and i[th] elements of vectors p and q respectively.
Design an algorithm to do matrix-vector multiplication **Ax=y**. Design an $O(N)$ algorithm to do this.
Hint:

$$a_{ij} = (p_j - q_i)^2 = q_i^2 - 2p_j q_i + p_j^2$$

$$y_i = (\sum_{j=1}^{N} x_j) q_i^2 - 2q_i (\sum_{j=1}^{N} p_j x_j) + (\sum_{j=1}^{N} p_j^2 x_j)$$

You must write a pseudocode showing line numbers and explain the number of computational steps with the help of line numbers. **(6 points)**

3. Shown below are the implementations of two possible ways of allocating a matrix. Which one (left or right) would you prefer for efficiency and why? No points awarded without explanation. **(4 points)**

```
B = (double **)malloc(m*sizeof(double *));
if (B == NULL) return NULL;
for (i = 0; i < m; i++) {
 B[i] = (double *)malloc(n*sizeof(double));
 if ( B[i] == NULL ) { free(B); return NULL; }
}
```

```
B = (double **)malloc(m*sizeof(double *));
if (B == NULL) return NULL;
B[0] = (double *)malloc(m*n*sizeof(double));
if (B[0] == NULL) { free(B); return NULL; }
/* now set the other pointers */
for (i = 1; i < m; i++)
B[i] = B[0] + i*n;
```

4. Shown below are two functions: one that uses function templates and one that doesn't. In a slight deviation from the template function example discussed in the class, the `MultiplyBy` template function shown here parametrizes a data element (in class we saw parametrizing of data types). Assume that there are no compilation or other errors. Which one would execute faster and why? No points awarded without explanation. **(4 points)**

```
int Multiply(int x, int m) {
      return x * m;
}

template <int m>
int MultiplyBy(int x) {
      return x * m;
}
```

```
//example usage.
int a, b;
a = Multiply(1000000,8);
b = MultiplyBy<8>(1000000);
```

5. Answer the questions a) to h) w.r.t. the code snippet shown. Assume that you have answered the previous question correctly while answering a question. **(12 points)**

```
class Ball {
private:
          const float pi;
          int radius;
public:
          Ball(int r) { radius = r;}
          void SetRadius(int rad);
          const Ball& operator=(const Ball& b);
          virtual void Print() {
              cout << radius << endl;
          }
};

class BasketBall: public Ball {
private:
          int color;
public:
          BasketBall(int radius, int c) {
              color = c;
          }
          void Print() {
                    cout << color << endl;
          }
};
```

a) Use member initialization list to initialize `pi` with `3.1415`. Show the modified implementation of the constructor. (1 point)

b) Implement the copy constructor of `Ball`. You must show the member function declaration and definition. (1 point)

c) Show the overloaded implementation of += operator such the following code snippet increments `radius` by 10. `Ball b; b+=10;` (1 point)

d) Complete the definition of member function `SetRadius` (1 point)

e) What should the copy assignment operator return so that the following code behaves as expected? (2 points)

```
Ball a(1), b(2), c(3);
a=b=c;

const Ball& Ball::operator=(const Ball& b) {
radius = b.radius;
____________________; // answer goes here.
}
```

f) Show how you would modify the `BasketBall` constructor so that it correctly initializes the `Ball` part of a `BasketBall` object. (2 points)

g) What would the following code snippet print? (2 points)

```
BasketBall a(1,100); Ball* b = &a;
a.Print(); b->Print();
```

h) Implement the destructors of `Ball` and `BasketBall` (2 points)

Part II – take home: (4 points) Visit the discussion forum to receive the question paper and instructions.