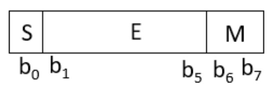
CS601: Software Development for Scientific Computing

Maximum Points: 25 | Mid-semester examination | 24/09/2021, 10:00AM to 12:00PM

Instructions:
This exam has two parts. Part I is open book, open notes (printed/written). No electronic devices allowed. Part II is take home. The instructions for part II are given separately in the README file provided along with the GitHub repository. State your assumptions (if any) clearly.

Part I:

1. Suppose the size of `float` is 1 byte in your machine and the format is as shown below (S=Sign, E=Exponent, M=Mantissa). Compare the largest non-integer number with the largest signed integer number possible with 1 byte of storage? You must explain your answer. **(2 points)**



2. You visit a supermarket and find that frequently accessed grocery items are stacked at the front aisles. Name the Computer Science concept that resonates with this. You must explain your answer. **(1 point)**

3. Tools
   i) The following `Makefile` will print three commands when you run `make` (assuming the programs have no syntax error or warning). Write the commands in the exact order in which they are printed. **(3 points)**

```
1  CXX=g++
2  CFLAGS=-I./inc
3  ifeq ($(DEBUG), 1)
4  CFLAGS += -g
5  endif
6
7  Solution: RDomain.o GridFn.o Solution.cpp
8          $(CXX) $(CFLAGS) $^ -o $@
9  GridFn.o: GridFn.cpp
10         $(CXX) $(CFLAGS) -c $< -o $@
11 RDomain.o: RDomain.cpp
12         $(CXX) $(CFLAGS) -c $< -o $@
13 clean:
14         rm -f *.o Solution
```

   ii) Suppose you change $^ at line 8 to $<. What would you expect? **(0.5 points)**

   iii) Suppose you want to use the target generated, `Solution`, with a debugger. What is the `make` command that you would issue to generate debug symbols? **(0.5 points)**

   iv) Consider the program below and the gdb session for the same shown adjacently. Fill in the blanks for A, B, C, and D shown in the gdb session. **(2 points)**

```
1 #include<iostream>
2 int fact(int n){
3  if(n<0)
4      return -1;
5  if(n==0)
6      return 1;
7  return n*fact(n-1);
8 }
9
10 int main() {
11  std::cout<<fact(5);
12 }
```

```
(gdb) b fact
(gdb) r
(gdb) c
(gdb) c
(gdb) c
(gdb) bt
#0  fact (n=2) at fact.cpp:3
#1          A          in fact (n=D) at fact.cpp:  B
#2          A          in fact (n=C) at fact.cpp:  B
#3  0x00000001004010b2 in fact (n=5) at fact.cpp:7
#4  0x00000001004010d3 in main () at fact.cpp:11
(gdb)
```

4. C++ Programming **(1 point)**

```
#include<iostream>
class B {
public:
        B( ) { }
        ~B() { }
        void print() {
                cout << "B::print( ) ";
        }
        void print(int i) {
                cout << "B::print(int)" << endl;
        }
};

class D : public B {
public:
        D( ) { }
        ~D() { }
        void print() {
                cout << "D::print( ) ";
        }
        void print(int i) {
                cout << "D::print(int)" << endl;
        }
};

int main(int argc, char * argv[]) {
        D *d = new D( );
        B *b = (B*) d;;
        b->print( );
        d->print(4);
        return 0;
}
```

What is printed?

a) D::print( ) D::print(int)

b) B::print( ) D::print(int)

c) D::print( ) B::print(int)

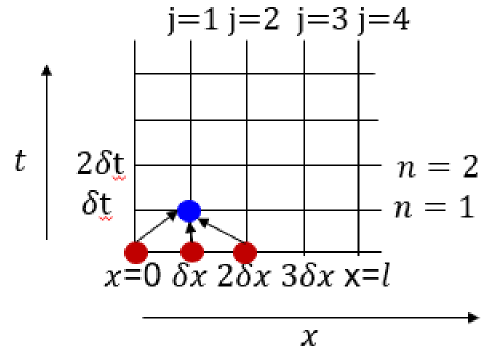
d) B::print( ) B::print(int)

e) Compiler error because it is ambiguous which B and/or D is to be called

5. Structured and Unstructured grids

i) The picture below shows a snapshot of the stencil computation for the 1D heat diffusion problem with Dirichlet boundary conditions. The dot at (j=1, n=1) indicates *part of the computation* to be done at time step 1 (i.e. at time δt from the beginning) using the data values from the initial conditions (shown using dots with arrows pointing to (j=1, n=1)).

Using the notation discussed in class, write the equations representing the *entire computation done at time step 3* (i.e. at time $3\delta t$ from the beginning). Assume that $l/\delta x = 4$. **(2 points)**



ii) The system of equations obtained while computing the 6-point stencil (Crank-Nicholson scheme) for the problem mentioned previously can be expressed in $Ax = B$ form. Using the notation discussed in class, a) Write the set of equations that you obtain *while computing at time step 2 (i.e. time= $2\delta t$ from the beginning)*. (2 points) b) Express the set of equations in $Ax = B$ form. (1 point) c) Write the A matrix obtained *while computing at time step 4*. Assume that $r = \alpha\delta t/(\delta x)^2 = 1.2$ (1 point) **(4 points)**

iii) Write briefly comparing the computation involved in 3-point stencil and 6-point stencil w.r.t. parallelism. E.g. parallelizing the computation at time step 1 would mean simultaneous computation of the dots at (j=1, n=1), (j=2, n=1), and (j=3, n=1). **(1 point)**

iv) Exploiting locality and parallelizing Delaunay triangulation is a challenge when compared to that in the 3-point stencil computation. Why or Why not? Justify your answers separately for locality and parallelism. **(2 points)**

Part II: **(6 points)**

Modify your PA2 implementation to compute the 2D Heat Equation in steady state. Your modification must incorporate the principles of Object-Orientation and have minimal changes w.r.t your PA2 code.

- You must create a branch in your GitHub repo called PA3
- You must tag your submission as `cs601pa3submission`
- You must submit (on GitHub) the tagged version of your code before Noon, 25/9/21.