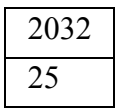
COSC 251 – Exam #1
Spring 2008
KEY

1.) Types and typing, control structures, scoping, expression resolution, error handling, data passing, concurrency, abstraction.
2.) A language is weakly typed if it allows for implicit casting of constants and variables of one type to another.
3.) FORTRAN, John Backus, IBM
4.) LISP, John McCarthy, MIT
5.) higher order functions, pure functions, recursion, lazy evaluation
6.) (λx.λy.x+y)
7.) C was named C due to its progression from the B programming language. B was a stripped down version of BCPL and was replaced by C due to B's lack of support for the newest flavor of UNIX.
8.) declare
fun{Sum N}
if N == 1 then 1 else N + {Sum N-1} end
end
9.) Function prototypes, void pointers, preprocessing
10.) int main(int argc, char **argv)
11.) if, while, switch, do while, for (the examples I leave to you)
12.) * - dereferencing/multiplication
& - referencing
++ - increment
13.) There is no difference between pass by copy and pass by value.
int foo (int a, int b).
14.) int, long, char, double, float, void, short
15.) private, public, protected
16.) ten 0's on individual lines
17.) A deep copy is a copy that copies all of the parameters from one object to another. A shallow copy only moves the pointer from one object to another. Copy constructors and overloaded equals are typically used to provide this functionality.
18.) 25. The memory diagram will look like (2000 is the starting addy for the top spot, 2032 is the starting addy for the bottom):



19.) no int returned for either foo or main, new is not a keyword in C, i is not declared.

20.)

```
//header file – circle.h

class circle{
        public:
                circle();
                circle(int);
                int getr ();
                void setr(int);
                float area();
                float circumference();
        private:
                int radius;
};

//class file – circle.cpp
#include “circle.h”
#include <math.h>

circle::circle(){ radius = 5; }

circle::circle(int r) { radius = r; }

int circle::getr() { return radius; }

void circle::setr(int r) { radius = r; }

float circle::area() { return (r*r*M_PI); }

float circle::circumference() { return (2*r*M_PI); }
```