CS 336 – Assignment 8

1. Describe three approaches to exception handling in languages that do not provide direct support for it. ***10 PTS***

2. In languages without exception-handling facilities, it is common to have most subprograms include an "error" parameter, which can be set to some value representing "OK" or some other value representing "error in procedure." What advantage does a linguistic exception-handling facility like that of Java have over this method? ***10 PTS***

3. Consider the following C++ skeletal program:

```
class Big {
  int i;
  float f;
  void fun1() throw int {
    . . .
    try {
      . . .
      throw i;
    . . .
    throw f;
    . . .
  }
  catch(float) { . . . }
 . . .
 }
}
```

```
class Small {
 int j;
 float g;
 void fun2() throw float {
  . . .
  try {
   . . .
   try {
    Big.fun1();
    . . .
    throw j;
    . . .
    throw g;
    . . .
  }
   catch(int) { . . . }
   . . .
  }
  catch(float) { . . . }
 }
}
```

In each of the four throw statements, where is the exception handled? Note that fun1 is called from fun2 in class Small. ***10 PTS***

4. What does the following Scheme function do? ***10 PTS***

```
(define (y s lis)
 (cond
  ((null? lis) '() )
  ((equal? s (car lis)) lis)
  (else (y s (cdr lis)))
))
```

5. What does the following Scheme function do? ***10 PTS***

```
(define (x lis)
 (cond
  ((null? lis) 0)
  ((not (list? (car lis)))
    (cond
     ((eq? (car lis) #f) (x (cdr lis)))
     (else (+ 1 (x (cdr lis))))))
  (else (+ (x (car lis)) (x (cdr lis))))))
```