1 Data

1. Describe the storage categories and be able to identify which variables hold data from each.
   - Static
   - Automatic
   - Dynamic

2. Compare and contrast reference-counting vs. mark-and-sweep garbage collection

3. Describe primitive (and not-so-primitive) data types
   - Integers
     (a) byte, short, int, long, long long
     (b) Java’s BigInteger
   - Real
     - Floating Point
     - Decimal (as in binary-coded decimal)
     - Rational

4. Explain the properties of
   - Strings
     - Java: pointer, offset, length
     - C: pointer, null-terminated
   - Arrays
   - Dynamic arrays
     - Java: ArrayList
     - C++: std::vector
     - C++: std::deque
     - persistent vector
   - Associative Array
     - C++: std::map
     - C++: std::unordered_map
   - Record
     - Be able to determine layout in memory

<table>
<thead>
<tr>
<th></th>
<th>short</th>
<th>byte</th>
<th>unused</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>int</td>
<td></td>
<td></td>
</tr>
<tr>
<td>byte</td>
<td></td>
<td>unused</td>
<td></td>
</tr>
<tr>
<td></td>
<td>int</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2 Scope

Be able to answer questions like the following

```c++
#include <iostream>
using namespace std;

int x = 100;
int y = 200;

void g() {
    cout << "g: y = " << y << endl;
    y = 20;
    cout << "g: y = " << y << endl;
}

void f() {
    cout << "f1: x = " << x << endl;
    cout << "f1: y = " << y << endl;
    int x = 1;
    int y = 2;
    g();
    cout << "f2: x = " << x << endl;
    cout << "f2: y = " << y << endl;
}

void h(int y) {
    cout << "h: x = " << x << endl;
    cout << "h: y = " << y << endl;
}

int main() {
    int x = 1000;
    int y = 2000;
    f();
    g();
    h(x);
    return 0;
}
```

a. What output is produced if we assume static (lexical) scope?

b. What output is produced if we assume dynamic scope?
3 Closures, Iterators and Generators

Be able to determine the output for code like these Python examples

```python
1 def make_generator():
2     a = 0
3     b = 1
4     def fun():
5         nonlocal a, b
6         next = a + b
7         a = b
8         b = next
9         return a
10     return fun
11
generator = make_generator()
12 for i in range(5): print(generator())

1 def powers(base):
2     value = 1
3     while True:
4         yield value
5         value *= base
6
7 p = powers(3)
8 for i in range(5): print(next(p))

1 def uniq(input):
2     seen = []
3     for element in input:
4         if element not in seen:
5             yield element
6             seen.append(element)
7
8 for e in uniq(['a', 'b', 'c', 'a', 'something', 1, 'b', 2, 1]): print(e)
```
4 Storage Classes

```c
#include <stdio.h>
#include <stdlib.h>

int G = 1;
int* P;

void func(int* arg1, int arg2) {
    static int S = 10;
    printf("S: %d, B: %d, C: %d\n", S, *arg1, A[0]);
    S = arg2;
}

int main() {
    int M;
    P = malloc(10*sizeof(int));
    M = 100;
P[0] = 200;
func(&M, P[0]);
func(P, G);
}
```

1. What does the program print?

2. Circle (below) all variables for which memory for the variable itself is allocated at compile time.
   G P S A M

3. Circle (below) all variables for which memory for the variable itself is allocated on the stack at runtime.
   G P S A M