Math 480a: Sage, Spring 2011, Homework 3

William Stein

Due April 20, 2011

Instructions: Do the following problems, and turn them in by the beginning of class on Wednesday, April 20, 2011. If you get help from a classmate or friend on any homework problem, please explicitly thank them in your write up.

These problems will mostly help you develop your knowledge of Python and Cython.

1. Below I've defined ten objects $a, b, c, d, e, f, g, h, i, j$ in Sage. Which are immutable, meaning the object itself can't be changed from Sage (assuming crazy tricks using Cython are not allowed)? For which objects `x` below does `hash(x)` not give an error message?

```
sage: a = "this is a string"
sage: b = (2,3,[4,5])
sage: c = set([1,2,3])
sage: d = {'a':1, 'b':3/4}
sage: class e: pass
sage: f = e()
sage: g = pi
sage: h = (-1,0,1)
sage: i = lambda x: x*x
sage: j = 2/3 + 1.5
```

2. As you hopefully found in Problem 1 above, there is supposed to be a `hash` method defined on objects precisely when they are immutable. Python assumes this in implementing various data types such as dictionaries, and bad things can happen if you violate this assumption (which you can).

   (a) Create a new Python class `hset` that derives from the builtin `set` type, but adds a new special method `__hash__`.

```
def __hash__(self):
    ???
```

   (b) Give an example of a Python dictionary with `hset` instances as keys.

   (c) Illustrate something that goes horribly wrong (e.g., could easily lead to subtle bugs) as a result of being able to make dictionaries with `hset` instances as keys, thus illustrating that it is a bad idea to define a `__hash__` method for mutable objects.

3. (a) Write a function `foo1` in Python that takes as input an int n with $1 \leq n \leq 2^{31} - 1$ and returns the power of 5 that exactly divides the integer. (You do

not have to validate the input; you may assume the input is in the above range, and if it isn't then the behavior of your function is "undefined".) For example, `foo1(15)` returns 1 and `foo1(875)` returns 3.

(b) Write a function in Cython called `foo2` with the same input and output as `foo1`.

(c) Race your functions. Use enough tricks so that `foo2` is at least 10 times as fast as `foo1` when given 875 as input.

4. Two creative problems:

(a) Describe in a paragraph two potential ideas for something you might want to do a final project on in this course. (You will be graded on whether you coherently describe ideas for two projects.)

(b) Describe a question you think would be good for me to put on the take-home midterm. (You will be graded based on whether you give a meaningful question that is relevant to the course.)