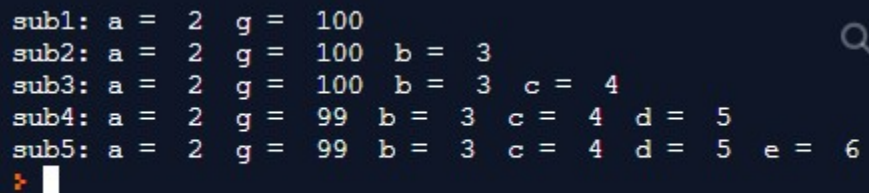


- 1 Static scope means that the object is bound at load time. This is in contrast to dynamic scope which is bound at run time of the code. The simplest example is in java, you use the keyword “static” to make something bound at load time. Without making a method static, your main method cannot use that method. An example of dynamic scope would be having a non-static method in java that you can only use when you dynamically create an object that contains that method.
3 Programming languages that support static scope: Java, Pascal, C++
3 Programming languages that support dynamic scope: Java, bash, LaTeX
-

- 2 Python subprogram can reference variables of their parent programs and global variables. This is done through using nonlocal and global. Simply use “nonlocal [variable name]” and the following sub program will be able to use the parent’s variable. This works exactly the same with global variables except using the keyword “global” instead. Subprograms can actually still read global variables without specifying, but if they want to modify the variables then it needs to be explicitly stated in this way.



```
sub1: a = 2 g = 100
sub2: a = 2 g = 100 b = 3
sub3: a = 2 g = 100 b = 3 c = 4
sub4: a = 2 g = 99 b = 3 c = 4 d = 5
sub5: a = 2 g = 99 b = 3 c = 4 d = 5 e = 6
```

- 3 Both malloc/free and new/delete are used to allocate/deallocate memory. New/delete is the newer version and has some much nicer attributes. An important one is that it never returns NULL (unlike malloc/free), instead it will throw an exception when something fails. This allows the code to more gracefully handle problems. New/delete do not have to specify the size in bytes of memory. The compiler will determine what size is needed based on the type of variable used, instead of how malloc/free needs to specify exactly how many bytes to allocate. Another really useful part of new/delete is that they use the objects constructor/destructor which allows for much better code.
-


- 4 Java, C#, and C++ all support pattern-matching using regular expressions with strings. This is typically used for things like `split()` that splits a string into substrings based on the given pattern. Java is different than C in that it stores strings in the `String` class. C uses char arrays for strings that are terminated by the null character. This changes what operations can be done on the strings. Java can use any methods inside the `String` class, but C uses array operations on the char array instead. Typically this doesn't make a huge difference, but sometimes java is more convenient than C.
-

- 5 Different operators have either left or right associativity in C. Some of the operators with left associativity are `*`, `/`, `%`, binary `+`, binary `-`. Some of the operators with right associativity are `++`, `--`, unary `-`, unary `+`. The most common binary operators are typically the ones with left associativity, the right associative operators all only operate on one variable. There is also a hierarchy of which operators occur first. For the binary operators the order is parentheses $>$ `*`, `/`, `%` $>$ `+`, `-`. If there are multiple operators of the same “tier” they are simply evaluated left to right (since they are left associative).

`Cout << (10 * 2 + 3) << endl;`

Results in 23 because `10 * 2` is evaluated before `+` 3.

```
1 #include <iostream>
2
3 int main() {
4     std::cout << (10 * 2 + 3) << std::endl;
5 }
```



```
./main
23
```

- 6 Technically all Java objects are explicit (fixed) heap-dynamic variables. Java then uses its garbage collector to deallocate these variables when they are no longer in use by the program. This allows for a very safe and convenient use as the programmer does not have to worry about deallocation because it is always done automatically. This is in contrast to C++ where not all variables are like this. Heap-dynamic variables can only be accessed through pointers, and they must be deallocated manually. This combines to make them much less reliable and convenient than java.
-

- 7 Java operands are always evaluated left to right. The left operand for binary operators is always fully evaluated before the right operand. This means sometimes the right operand will modify the left operand, but it will be after the left operand is already evaluated.

```
int[] a = {4,4};
int b = 1;
a[b] = b = 0;
```

The last line in this code is where the operand evaluation order is important. First the left operand “a[b]” is evaluated to a[1], then right operand “b = 0” is evaluated to simply 0. This results in the code setting a[1] = 0 and b = 0.

```
3 int[] a = {4,4};
4 int b = 1;
5 a[b] = b = 0;
6 System.out.println("a: ["+a[0]+", "+a[1]+"]");
7 System.out.println("b: "+b);
8 }
9 }
```

```
a: [4, 0]
b: 0
[]
```

8 How is Python an interpreted language?

Python is an interpreted language because the code that people write goes through an interpreter for the computer processor to understand.

What is the difference between Python Arrays, lists, tuples, and records? Explain it with examples

xxx

What does [::-1] do? Explain it with an example

If an array A exists, A[::-1] will loop through all elements of the array backwards.

How can you randomize the items of a list in place in Python?

Randomizing the items in a list in place in python is easiest when using the random module. You can simply call shuffle() which will randomize the original list in place.

What is the difference between range & xrange? Explain it with an example

The main difference between range and xrange is their return type. Range returns a list, and xrange returns an xrange object.

What advantages do NumPy arrays offer over (nested) Python lists?

They take up less space, they are faster and offer higher performance, and they have more built in functions available.

How to add values to a python array? Explain it with an example

There are two main ways to add values to an array. Array.append(item) adds one item. Array.extend([a, b, c]) adds multiple items.

What is split used for? Explain it with an example

Split is used to split a string into smaller substrings.

```
txt = (" tea, coffee, milk")
```

```
a = txt.split(",")
```

“a” now contains 3 strings, “tea”, “coffee”, and “milk”