Worksheet 2: CS121

SI Arnesh

Review Notes on What We've Been Studying So Far:

What does returning a variable actually mean? Let's look at an example.

```
public int returnSum(int a, int b)
{
    return a+b;
}
```

So we all know, the *method* returnSum returns the value of a + b. But what does this mean? What do we need it for?
This has to do with calling the method. When calling a method we use an object and the method Name.

If we want to add 69280 and 49210 using this method, we use an object we have created, say 'popcorn', and call the method plugging in the two values for a and b.

```
popcorn.returnSum(69280, 49210);
```

Back to our question regarding return . When we use the above statement, where does the returned value go to? The answer is nowhere, since we don't allocate memory space to store the returned value. How do we do this? Simple. Same as how we allocate memory space to receive data normally.

```
int sum = popcorn.returnSum(69280, 49210);
```

The variable 'sum' now stores the value 118490, which is *returned* by the *function* returnSum, *called on* the *object* popcorn, and stored in the *variable* sum.

Note: Difficulty is indicated by number of '*' signs next to the question. There could be multiple correct answers for every question.

1) System.out.print("6+0.5");

   What is the *output*? *

   A) 60.5
   B) 6.5
   C) Compiler error
   D) 6+0.5

2) int num1 = 50.0;
   double d = 5;
   int num2 = num1/d;

   What value does *num2* store? **

   A) 10
   B) 10.0
   C) 0
   D) Compiler error

3) int num1 = 1;
   int num2 = 2;
   int num3 = 1;
   num2*=num2;
   num1+=num3+num1;
   num3 = num1;
   num3-= num2;
   System.out.print(num3);

   What is the output? **

   A) 4
   B) 3
   C) 1
   D) -1

4) Consider the following simple Java program:

```
import java.util.*;
public class Tree
{
        private String name;
        private int age;

        public Tree(String name, int age){
                this.name = name;
                this.age = age;
        }

        public void setName(String name){
                this.name = name;
        }
        public void setAge(int age){
                this.age=age;
        }

        public void takeInput(){
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter name of your tree.");
                setName(sc.next());
                System.out.println("Enter age of your tree.");
                setAge(sc.nextInt());
        }

        public String getName(){
                return name;
        }
        public int getAge(){
                return age;
        }

        public void displayDetails(){
                System.out.println(" This tree is called " + getName() + " and it is "+getAge() +"
years old");
        }

        public static void main(String args[]){
                // TO-DO
        }
}
```

A) Let us implement the main method.

i) Which of the following creates an object of Tree class? **

a) Tree myObject = new Tree();
b) Tree myObject = Tree("Banyan", 250);
c) Tree myObject = new Tree("Banyan", 250);
d) Object myObject = new Tree();

ii) Now print the name. *

a) System.out.print(getName());
b) System.out.print(myObject.getName());
c) System.out.print(name);
d) System.out.print(obj.getName());

iii) It has been a year since you updated your object! Change age to 251. *

a) age = 251;
b) setAge(251);
c) myObject.setAge();
d) myObject.setAge(251);

iv) Your tree has been felled by corporate deforestation! In your grief you decide to store a new tree in the object. Take new data values as input from the user. *

a) takeInput();
b) myObject.takeInput();
c) myObject.setName(); myObject.setAge();
d) obj.takeInput();

v) Display the details of your object. *

a) obj.displayDetails();
b) displayDetails();
c) myObject.displayDetails();
d) myObject.displayDetails;

B) What is a default constructor? *

C) List all the *accessors* of the class. **

D) List all the *mutators* of the class. ***

E) List all other *methods* of the class. **

F) List all the *data members* of the class. *

G) Name all the *objects* in the class. ***

5) Which of these methods cause compiler errors (and why)? ***

a)
```
int x(int y)
{
        return 5.0;
}
```

b)
```
String str()
{
        return 'abc';
}
```

c)
```
int a(int b)
{
        return a/b;
}
```

d)
```
char c()
{
        return "c";
}
```

e)
```
double D(int z)
{
        return z+3;
}
```

f)
```
String str2()
{
        return 5.6 + "years";
}
```

g)
```
void print()
{
        return;
}
```

h)
```
int name()
{
        return;
}
```