Day 2

1. What is stack machine? Why stack machine is important to .NET?
2. What is the different between heap and stack in .NET?
3. What is Virtual memory ?
4. What conditions will cause Garbage Collector run?
5. What is unsafe code in C#? explain how can you use unsafe code in C#.
6. What are disadvantages of unsafe code?
7. Explain the advantages of tail recursion to non-tail recursion?
8. Write a program that reads in a sequence of characters and prints them in reverse order. Use a stack
9. Write a program that reads in a positive integer and prints the binary representation of that integer. Hint: divide the integer by 2
10. Given the following instructions to evaluative a postfix expression “2 3 1 \* + 9 -“

1)Scan ‘2’, it’s a number, so push it to stack. Stack contains ‘2’  
2) Scan ‘3’, again a number, push it to stack, stack now contains ‘2 3’ (from bottom to top)  
3) Scan ‘1’, again a number, push it to stack, stack now contains ‘2 3 1’  
4) Scan ‘\*’, it’s an operator, pop two operands from stack, apply the \* operator on operands, we get 3\*1 which results in 3. We push the result ‘3’ to stack. Stack now becomes ‘2 3’.  
5) Scan ‘+’, it’s an operator, pop two operands from stack, apply the + operator on operands, we get 3 + 2 which results in 5. We push the result ‘5’ to stack. Stack now becomes ‘5’.  
6) Scan ‘9’, it’s a number, we push it to the stack. Stack now becomes ‘5 9’.  
7) Scan ‘-‘, it’s an operator, pop two operands from stack, apply the – operator on operands, we get 5 – 9 which results in -4. We push the result ‘-4’ to stack. Stack now becomes ‘-4’.  
8) There are no more elements to scan, we return the top element from stack (which is the only element left in stack).

Using class Stack in C# to evaluate the previous expression

1. What is the output of this “5 9 3 + 4 2 \* \* 7 + \*”
2. Write a recursive function to find min of an array
3. Write a recursive function to find out the Fibonacci of N
4. Change the previous exercise to tail recursion; write down the value of the function after each call(tracing)