1.How do you run a program in gdb?
If ./a.out is a executable file which we want to debug . To run a program in gdb , first user must launch the debugger by typing gdb ./a.out . After this step , gdb will be waiting for the user to type the command . User should ype run to run the program in gdb.

2.How do you pass command line arguments to a program when using gdb?
The user can give the command line arguments to ./a.out the same way as on the command line in unix except that the user must say run instead of ./a.out (ie filename) .

3.How do you set a breakpoint in a program?
break function sets the breakpoint at the beginning of function.If the code is in multiple files,the user must specify filename : function.
break linenumber or break filename:linenumber sets the breakpoint to the given line number in the source file.

4.How do you set a breakpoint which only occurs when a set of conditions is true (eg when certain variables are a certain value)?
Let flag be a variable . If the user wants to break at breakpoint 1 only if flag is set to 1 . The user has to type
condition 1 flag == 1

5.How do you execute the next line of C code in the program after a break?
By typing step or next and then hitting enter

6.If the next line is a function call, you'll execute the call in one step. How do you execute the C code, line by line, inside the function call?
By typing step and typing enter.

7.How do you continue running the program after breaking?
By typing continue and then hitting enter

8.How can you see the value of a variable (or even an expression) in gdb?
type print variable_name and hitting enter

9.How do you configure gdb so it prints the value of a variable after every step?
By typing display variable_name and hitting enter

10.How do you print a list of all variables and their values in the current function?
typing i lo and hitting enter will give all local variables
typing i var and hitting enter will give all global and static variables

11.How do you exit out of gdb?
typing quit and hitting enter.

Exercise : 2

Bug in AppendTest.c

In C , the strings end with delimiter '\0' . When printing a string in c,

the string is printed till the '\0' character is encountered . The bug in the program is that the '\0' character of the second string is not copied
int the first string . Hence when running multiple times , a part of the previous string is also printed . The modification is the for statement . It should be
for (k=0; k<=s2len; k++) {
instead of
for (k=0;k<s2len;k++){

Exercise : 3

The program gave segmentation fault because the of the scanf statement .
It should have been
scanf("%d",&input);
instead of
scanf("%d",input);

After the above modification , it was observed that the sum was 0. Hence the average was printed to be zero . This is because the read_values function is not communicating the value of the sum back to main . The code was modified so that the main passes the reference of the variable sum to the read_values function which updates the value in that address .

The code should be :

```
int read_values(double *sumaddr)
{
  int values=0,input=0;
  //sum = 0;
  printf("Enter input values (enter 0 to finish):\n");
  scanf("%d",&input);
  while(input != 0) {
    values++;
    *sumaddr += input;
    scanf("%d",&input);
  }
  return values;
}

int main()
{
  double sum=0;
  int values;
  values = read_values(&sum);
  printf("Average: %g\n",sum/values);
  return 0;
}
```

Programming for Performance
Lab Session 1: Catch the Bugs.

No Type of bug: But there is a possibility of reachable memory leak if the caller function does not free

the char * pointer that has been returned from the function .

Lab Session 2: Catch the Bugs.

YES Type of bug: Memory Leak (definitely lost) for the node created using malloc

Lab Session 3: Catch the Bugs.

No TYPE of bug: No bug