Appendix

```
// program to generate normally distributed random variables
 user input //
<============================== random number seed //
<======================================== mean //
<================================ standard deviation //
<======================== number of samples to generate //
------------------------------------------------------------------------------ //
----------------------------include file-----------------------------------//
()include<stdio.h>           // Need for pintf#
()include<stdlib.h>          // Need for exit#
()include<math.h>            // Need for sqrt() and log#
 --------------------------------Defines-------------------------------------//
define PI                    3.142857#
----------------------Function prototypes---------------------------------//
double norm(double mean,double std_dev); // Return a normal rv
;(double rand_val(int seed
        //=========Main program ========================
(void main(void
}
FILE *fp_out;                          // File pointer to
output file
char instring[80];                     // Input string
double num_samples;                    // Number of samples
to generate
double mean,std_dev;                   // Main and standard
 deiation
```

```
double norm_rv;                    // The adjusted normal vale
int i;                                  // Loop counter
;("printf("output file name
;(scanf("%s",instring
;("fp_out=fopen(instring,"w
(if(fp_out==NULL
}
;(printf("ERROR in creating output(%s)\n",instring
;(exit(1
{
Prompt for random number seedand thn use it //
        printf("Random nmber seed ============================>");
;(scanf("%s",instring
;((rand_val((int) atoi(instring
Prompt for mean value //
        printf("mean=====================================>");
;(scanf("%s",instring
;(mean= atof(instring
Prompt for standad dviation //
                printf("standard deviation ======================>");
;(scanf("%s",instring
;(std_dev= atof(instring
Prompt for number of samples to geerate //
printf("Number of samples to generate
;("<======================
;(scanf("%s",instring
;(num_samples= atoi(instring
```

```
(++for(i=0;i<num_samples;i
}
Generate a normally distributed rv //
;(norm_rv = norm( mean, std_dev
Output the norm_rv value //
;(fprintf(fp_out,"%f\n",norm_rv
{
Output message and close the disribution and output files //
;("printf("-------------------------------------------------\n
;("printf("-Done!\n
;("printf("-------------------------------------------------\n
;(fclose(fp_out
{
            // ============================================
function to generate nrmally dustributed random variable using --- //
box muller method
//  --- Input: mean and standard deviatio
n                                          // -- -Output: Returns with normally
                    distribted random variables
         //============================================
(double norm(double mean,double std_dev
}
double u,r,theta;          // Variabes for Box-Muller method
  double x;                // Normal(0,1) rv
double norm_rv;            // The adjuste normal rv
Generate u //
;u = 0.0
```

```
(while(u==0.0
;(u = rand_val(0
Compute r //
;((r = sqrt(-2.0*log(u
Generate theta //
;theta = 0.0
(while(theta==0.0
;(theta = 2.0*PI*rand_val(0
Generate x value //
;(x = r * sin(theta
Adjust x vlue for specified mean and variance //
;norm_rv = (x*std_dev)+mean
Retrn the normally distributed RV value //
;(return(norm_rv
{
        //===========================================
(double rand_val(int seed
}
;const long a= 16807
;const long m= 2147483647
;const long q= 127773
;const long r= 2836
;static long x
;long x_div_q
;long x_mod_q
;long x_new
Set the seed if argument is non-zero and then reurn zero //
```

```
(if (seed>0
}
;x = seed
;(return(0,0
{
RNG using integer arithmetic //
;x_div_q = x / q
;x_mod_q = x %q
;(x_new = (a * x_mod_q) – (r * x_div_q
(if(x_new>0
;x = x_new
else
;x = x_new+m
Return a random value between 0.0and 1.0 //
;(return((double) x / m
                        {
```