

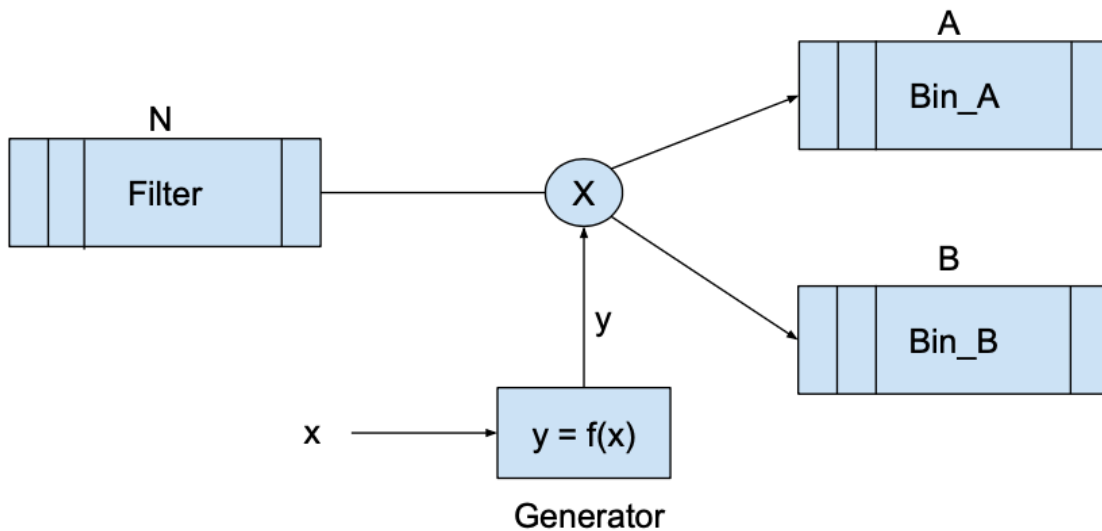
Instructions:

- You should save the program with the file name as specified as <Test#>-<Set#>-<Roll#>.c. For example, **01-A-24NA10006.c** to save Test1 of Set A with Roll Number 24NA10006.
- You should upload the program to the Moodle system. Also, copy + paste your program to the text window on the test page.
- There is one problem and the maximum time allowed is 120 minutes.

In this problem, you have to work with the following three 1-D Arrays. :

- Filter : An array of floating-point values of size N.**
- Bin_A : An array of integer values of size A.**
- Bin_B : An array of integer values of size B.**

A system with these arrays is shown in the figure given below.



Your task is to populate these arrays following the strategies as stated below.

1. Filling the filter `Filter`:

- a. Given a generator as shown below, which takes a value x and produces an output y .**

$$y = \sum_{k=1}^x \frac{k^2}{2k+1} - (-1)^k \cdot k + \sin\left(\frac{x}{3}\right)$$

- b. Generate a value x as a random number in the range [1 to 100].**
c. Compute y by using the above generator.
d. Repeat the above two steps N times to fill the array `Filter`.

2. Entering values into `Bin_A` and `Bin_B`:

- a. Generate a random number x in the range [1 to 100].**
b. Calculate y for the value x .
c. If y is smaller than any value in `Filter`, store x into `Bin_A`.
d. If y is greater than any value in `Filter`, store x into `Bin_B`.

3. Repeat Steps 2, until either `Bin_A` or `Bin_B` is full.

4. Display the values in the arrays.

Note:

1. You should define the values of N , A , B and other values in the generator as constant.
2. For the calculation of $\sin(x)$, include `math.h` and use the `sin()` function.
(Use ``gcc <Test#>-<Set#>-<Roll#>.c -lm`` to execute the program)
3. The code for random number generator in the range 1 to 100 is given below:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main() {
    // Random number generator with a fixed seed (42) to check
    // the test cases
    srand(42);
    int randomNumber = rand() % 100 + 1;
    printf("Random Number: %d\n", randomNumber);
    return 0;
}
```

```

// Code creator: Nishkal Prakash (nishkal@iitkgp.ac.in)
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define N 10
#define A 5
#define B 5

// Random number generator with a fixed seed (42) to check the test cases
int r() {
    return rand() % 100 + 1;
}

// Generator function
float f(int x) {
    float y = 0;
    for(int k = 1; k <= x; k++)
        y += (k*k)/(2*k+1) - pow(-1, k)*k;
    y += sin(x/3);
    return y;
}

// Main function
int main() {
    // Random number generator with a fixed seed (42)
    srand(42);
    // Defining the arrays
    float Filter[N];
    int Bin_A[A], Bin_B[B];
    // Loop variables and other variables
    int i, j, k;
    float y, x;

    // Filling the Filter array
    for(i = 0; i < N; i++)
        Filter[i] = f(r());

    // Computing the maximum and minimum values in Filter
    float maxFilter = Filter[0], minFilter = Filter[0];
    for(i = 1; i < N; i++) {
        if(Filter[i] > maxFilter)    maxFilter = Filter[i];
        if(Filter[i] < minFilter)    minFilter = Filter[i];
    }
    // printf("Max: %.2f, Min: %.2f\n", maxFilter, minFilter);
}

```

```

// Entering values into Bin_A and Bin_B
int A_len = 0, B_len = 0;
while(1) {
    // Loop until either Bin_A or Bin_B is full
    x = r();
    y = f(x);

    // if y > minFilter, store x in Bin_A
    if (y < maxFilter) Bin_A[A_len++] = x;
    if (y > minFilter) Bin_B[B_len++] = x;
    if(A_len == A || B_len == B) break;
}

// Printing the arrays
printf("Filter: [");
for(int i = 0; i < N-1; i++)
    printf("%.2f, ", Filter[i]);
printf("%.2f]\n", Filter[N-1]);

printf("\nBin_A: [");
for(i = 0; i < A_len-1; i++)
    printf("%d, ", Bin_A[i]);
printf("%d]\n", Bin_A[A_len-1]);

printf("\nBin_B: [");
for(i = 0; i < B_len-1; i++)
    printf("%d, ", Bin_B[i]);
printf("%d]\n", Bin_B[B_len-1]);

return 0;
}

/* Sample Output:
Filter: [1367.87, 1.00, 1154.15, 813.15, 1680.27, 1520.76, 72.04,
507.65, 15.14, 1154.15]

Bin_A: [10, 51, 72, 37]

Bin_B: [10, 51, 72, 100, 37]
*/

```