Задачи на машзал 13 октября 2015 г

mz06-1. Программе в аргументах командной строки задаются четыре целых 32-битных числа:

- count (count >= 0, count < 100000) – число чисел для генерации;
- low (low >= -32768);
- high (high <= 32768, low < high);
- seed (seed > 0).

На стандартный поток вывода напечатайте count псевдослучайных чисел в интервале [low, high), полученных с помощью функции rand() с начальным значением генератора, задаваемым seed. Для получения случайных чисел в заданном диапазоне используйте константу RAND_MAX.

В формуле R = U * RANGE + low, где U – равномерно распределенная на [0;1) случайная величина, RANGE – число целых чисел в интервале [low;high), R – искомое число, добавьте приведение к типу int в таком месте, чтобы формула была корректна и для положительных, и для отрицательных значений low и high.

mz06-2. В аргументе командной строки задается имя текстового файла, который содержит временные отметки. Входной текстовый файл состоит не менее чем из одной строки текста. Каждая строка текста не превышает в длину 1000 символов и содержит одну временную отметку в формате:
YYYY/MM/DD hh:mm:ss
перед YYYY, между DD и hh и после ss может находиться произвольное число пробельных символов. В остальном дата записывается точно в указанном формате.
На стандартный поток вывода напечатайте последовательность целых чисел, представляющую собой интервалы в секундах между последовательными временными отметками во входном файле.
Учитывайте переходы летнее/зимнее время.

mz06-3. Программе на стандартном потоке ввода задаются следующие параметры дискретной случайной величины.

- Количество различных значений (n), которые принимает случайная величина (0 < n <= 100).
- n пар целых чисел, первое из которых задает значение, принимаемое случайной величиной (32-битное знаковое целое число), а второе – вероятность данного значения в процентах.
- Количество чисел (m), которое необходимо сгенерировать (0 <= m < 1000).
- Затравка для ГПСЧ (seed).

С использованием алгоритма генерации случайных чисел с заданным распределением из равномерно распределенных случайных чисел, описанного ниже, сгенерируйте и напечатайте на стандартный поток вывода указанное число случайных чисел.

Так как вероятности выпадения значений задаются в процентах, то сначала необходимо сгенерировать случайную величину в интервале [0; 100). Интервал [0; 100) разбивается на подинтервалы, соответствующие процентной вероятности выпадения заданного значения случайной величины, в порядке их описания во входном файле. Например, если случайная величина принимает 3 значения с вероятностями 20%, 30% и 50%, то интервал [0;100) разбивается на подинтервалы [0; 20), [20; 50), [50; 100). В зависимости от того, в какой подинтервал попало равномерно распределенное случайное число, берется соответствующее значение требуемой случайной величины.

mz06-4. Модифицируйте решение задачи mz04-1 следующим образом: генерацию случайных чисел в диапазоне [low, high) реализуйте в виде отдельных функций

```
void rand_seed(int seed);           // инициализация ГПСЧ
int rand_range(int low, int high);
```

Функция должна находиться в отдельном файле rand.c, а необходимые типы данных и прототипы должны находиться в заголовочном файле rand.h. Эти функции должны использовать стандартные функции srand и rand.
Напишите файл genrand.c, который будет решать задачу mz04-1 с помощью вызова функций из rand.c. Напишите Makefile для сборки проекта. Исполняемый файл (результат сборки проекта) должен называться genrand, а каталог проекта, в котором находятся исходные файлы и Makefile и в котором выполняется компиляция – genrand.
В тестирующую систему должен сдаваться tar.gz архив каталога genrand, в котором должны присутствовать только исходные файлы. Объектные и исполняемые файлы должны быть удалены.

Замечание. Если каталог называется genrand, то архив genrand.tar.gz можно получить следующей командой из каталога, в котором находится каталог genrand:
tar cfz genrand.tar.gz genrand

mz06-5. Доопределите структуру RandomOperations необходимыми полями:

```
typedef struct RandomOperations
{
      //…
} RandomOperations;
```

напишите реализацию функции

```
RandomOperations *random_create(const char *param);
```

напишите реализации недостающих функций таким образом, чтобы фрагмент

```
RandomOperations *rr = random_create(“1234”);
for (int j = 0; j < 100; ++j) {
  printf(“%d\n”, rr->next());
}
```

выводил на печать 100 псевдослучайных чисел, полученных в результате вызова функции rand() с затравкой 1234.