Задачи:

1. Реализация на Hashtable с метода Linear Probing
2. Реализация на Hashtable с метода Separate Chaining
3. Троично-наредено дърво T с точност R (положително число с плаваща запетая в интервала (0,1)) наричаме структура, която се характеризира с корен със стойност x (число с плаваща запетая) и три троично-наредени поддървета: TL, TM и TR, за които:
   * всички елементи в TL са със стойности < x(1–R);
   * всички елементи в TR са със стойности > x(1+R);
   * всички елементи в TM са със стойности в интервала [x(1–R), x(1+R)].
   Всяко от поддърветата може да е празно. Разглеждаме структура TONode, която описва възел в троично-наредено дърво:

   Да се реализира функция bool insert(TONode *&root, double x, double R)
   за вмъкване на елемента x в троично-наредено дърво с коренов възел root и точност R така, че дървото да остане троично-наредено, като вмъкването се осъществява само ако x още не присъства в дървото. Функцията да връща true, ако операцията е успешна и елементът е вмъкнат и false, в противен случай.

4. “Разредена матрица” наричаме структура от данни, представяща матрица A(MxN), състояща се от елементи от произволен тип, чиито стойности могат да се инициализират с константата 0 и да се сравняват с операторите == и !=. С всеки елемент могат да се извършват операциите присвояване на стойност и четене на стойност. При конструиране по подразбиране, елементите на матрицата се инициализират с 0. Структурата от данни “разредена матрица” трябва да удовлетворява следните изисквания за сложност (с n е означен броя на ненулевите елементи):
   * Използваната от матрицата памет да расте линейно спрямо броя ненулеви елементи, т.е. структурата трябва да има сложност по памет O(n).
   * Сложността по време за достъп до елемент на матрицата да е най-много логаритмична, т.е. O(log n).

   Да се дефинира шаблон на клас SparseMatrix, представящ разредена матрица, който да поддържа следните операции:
   1. Метод isZeroRow(row), който да проверява дали всички елементи на даден ред от матрицата са нулеви.

   2. Ако m е матрица, да се поддържа конструкцията for(unsigned int i : m) {...}, с която могат да се обходят индексите на всички редове, в които има поне един ненулев елемент.
   Упътване: може да използвате помощен клас или обект за обхождане на ненулевите редове.

3. Метод set(i, j, value), който задава стойност value на елемента $a_{i,j}$.
4. Оператор за индексиране в константна форма, който по индекс на ред и колона дава достъп за четене на елемент, например a[i][j]. Ако позицията е невалидна, операторът да връща константата 0.
Внимание: тъй като операторът не променя стойности на елементи, извикването му не трябва да води до заделяне на допълнителна памет!
Упътване: може да използвате помощен клас, описващ ред на матрицата.
5. Да се дефинират оператори + и += за събиране на разредени матрици. Резултатът да е разредена матрица, отговаряща на изискването за сложност по памет.
Бонус: Да се предложат подходящи тестове за тези оператори.