Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра Программной Инженерии

Лабораторная работа 13

По дисциплине «Основы алгоритмизации и программирования»

На тему «Бинарные кучи»

Выполнил:

Студент 1 курса 3 группы

Шатерник Г.И.

Преподаватель: асс. Андронова М.В.

2023, Минск

5.  В проект добавить следующие функции: удаление минимального **extractMin**; удаление i-ого элемента **extractI**; объединение **unionHeap** двух куч в одну.

|  |
| --- |
| Программный код(lab13.cpp)  //Куча может быть реализована с использованием связанного списка в C++ путем определения структуры узла, которая содержит значение и указатель на следующий узел. Мы также можем определить класс кучи, который содержит указатель на головной узел и предоставляет методы для вставки и удаления узлов при сохранении свойства кучи.  #include <iostream>  using namespace std;  // Определите структуру узла  struct Node {  int value;  Node\* next;  };  // Определите класс кучи  class Heap {  public:  Heap() {  head = nullptr;  }  // Вставляем новое значение в кучу  void insert(int value) {  Node\* newNode = new Node{ value, nullptr };  if (!head) {  head = newNode;  return;  }  if (value > head->value) {  newNode->next = head;  head = newNode;  return;  }  Node\* curr = head;  while (curr->next && curr->next->value > value) {  curr = curr->next;  }  newNode->next = curr->next;  curr->next = newNode;  }  // Удалить и вернуть максимальное значение в куче  int removeMax() {  if (!head) {  cout << "Heap is empty." << endl;  return -1;  }  Node\* temp = head;  head = head->next;  int maxVal = temp->value;  delete temp;  return maxVal;  }  private:  Node\* head;  };  int main() {  Heap heap;  heap.insert(5);  heap.insert(3);  heap.insert(8);  heap.insert(2);  heap.insert(7);  cout << heap.removeMax() << endl;  cout << heap.removeMax() << endl;  cout << heap.removeMax() << endl;  cout << heap.removeMax() << endl;  cout << heap.removeMax() << endl;  cout << heap.removeMax() << endl;  return 0;  } |
| Программный код (Heap.cpp)  #include "Heap.h"  #include <iostream>  #include <iomanip>  namespace heap  {  int Heap::left(int ix)  {  return (2 \* ix + 1 >= size) ? -1 : (2 \* ix + 1);  }  int Heap::right(int ix)  {  return (2 \* ix + 2 >= size) ? -1 : (2 \* ix + 2);  }  void\* Heap::extractMax()  {  void\* rc = nullptr;  if (!isEmpty())  {  rc = storage[0];  storage[0] = storage[size - 1];  size--;  heapify(0);  } return rc;  }  int Heap::parent(int ix)  {  return (ix + 1) / 2 - 1;  }  void Heap::swap(int i, int j)  {  void\* buf = storage[i];  storage[i] = storage[j];  storage[j] = buf;  }  void Heap::heapify(int ix)  {  int l = left(ix);  int r = right(ix);  int irl = ix;  if (l > 0)  {  if (isGreat(storage[l], storage[ix]))  irl = l;  if (r > 0 && isGreat(storage[r], storage[irl]))  irl = r;  if (irl != ix)  {  swap(ix, irl);  heapify(irl);  }  }  }  void Heap::deleteHeap()  {  if (!isEmpty())  {  size = 0;  this->~Heap();  }  }  Heap create(int maxsize, CMP(\*f)(void\*, void\*))  {  return \*(new Heap(maxsize, f));  }  void Heap::scan(int i) const  {  int probel = 20;  int z = 3;  std::cout << '\n';  if (size == 0)  std::cout << "Куча пуста";  for (int u = 0, y = 0; u < size; u++)  {  std::cout << std::setw(probel + 10) << std::setfill(' ');  ((AAA\*)storage[u])->print();  if (u == y)  {  std::cout << '\n';  if (y == 0)  y = 2;  else  {  y = y + z;  z++;  }  }  probel /= 2;  }  std::cout << '\n';  }  int Heap::findMin()  {  int min = 0;  for (int i = 1; i < size; i++)  {  if (((AAA\*)storage[i])->getPriority() < ((AAA\*)storage[min])->getPriority())  min = i;  }  return min;  }  void\* Heap::extractI(int ix)  {  void\* rc = nullptr;  if (!isEmpty())  {  rc = storage[ix];  storage[ix] = storage[size - 1];  size--;  heapify(0);  } return rc;  }  void\* Heap::extractMin()  {  int min = findMin();  return extractI(min);  }  void Heap::insert(void\* x)  {  int i;  if (!isFull())  {  storage[i = ++size - 1] = x;  while (i > 0 && isLess(storage[parent(i)], storage[i]))  {  swap(parent(i), i);  i = parent(i);  }  }  }  }  void AAA::print()  {  std::cout << x;  }  int AAA::getPriority() const  {  return x;  } |
| Программный код (Heap.h)  #include "Heap.h"  #include <iostream>  #include <iomanip>  namespace heap  {  int Heap::left(int ix)  {  return (2 \* ix + 1 >= size) ? -1 : (2 \* ix + 1);  }  int Heap::right(int ix)  {  return (2 \* ix + 2 >= size) ? -1 : (2 \* ix + 2);  }  void\* Heap::extractMax()  {  void\* rc = nullptr;  if (!isEmpty())  {  rc = storage[0];  storage[0] = storage[size - 1];  size--;  heapify(0);  } return rc;  }  int Heap::parent(int ix)  {  return (ix + 1) / 2 - 1;  }  void Heap::swap(int i, int j)  {  void\* buf = storage[i];  storage[i] = storage[j];  storage[j] = buf;  }  void Heap::heapify(int ix)  {  int l = left(ix);  int r = right(ix);  int irl = ix;  if (l > 0)  {  if (isGreat(storage[l], storage[ix]))  irl = l;  if (r > 0 && isGreat(storage[r], storage[irl]))  irl = r;  if (irl != ix)  {  swap(ix, irl);  heapify(irl);  }  }  }  void Heap::deleteHeap()  {  if (!isEmpty())  {  size = 0;  this->~Heap();  }  }  Heap create(int maxsize, CMP(\*f)(void\*, void\*))  {  return \*(new Heap(maxsize, f));  }  void Heap::scan(int i) const  {  int probel = 20;  int z = 3;  std::cout << '\n';  if (size == 0)  std::cout << "Куча пуста";  for (int u = 0, y = 0; u < size; u++)  {  std::cout << std::setw(probel + 10) << std::setfill(' ');  ((AAA\*)storage[u])->print();  if (u == y)  {  std::cout << '\n';  if (y == 0)  y = 2;  else  {  y = y + z;  z++;  }  }  probel /= 2;  }  std::cout << '\n';  }  int Heap::findMin()  {  int min = 0;  for (int i = 1; i < size; i++)  {  if (((AAA\*)storage[i])->getPriority() < ((AAA\*)storage[min])->getPriority())  min = i;  }  return min;  }  void\* Heap::extractI(int ix)  {  void\* rc = nullptr;  if (!isEmpty())  {  rc = storage[ix];  storage[ix] = storage[size - 1];  size--;  heapify(0);  } return rc;  }  void\* Heap::extractMin()  {  int min = findMin();  return extractI(min);  }  void Heap::insert(void\* x)  {  int i;  if (!isFull())  {  storage[i = ++size - 1] = x;  while (i > 0 && isLess(storage[parent(i)], storage[i]))  {  swap(parent(i), i);  i = parent(i);  }  }  }  }  void AAA::print()  {  std::cout << x;  }  int AAA::getPriority() const  {  return x;  } |
| Вывод: |