Министерство образование Республики Беларусь

Учреждение образования «Полоцкий государственный университет»

Информационно-технологический факультет

Кафедра технологий программирования

Лабораторная работа №3 по теме «Перегрузка операции»

По дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Выполнил: Ролевич М.А.

Группа 19-ИТ-1.

Проверил: Васильева Д. М.

Новополоцк

2021 г.

**Вариант 3**

**Цель.** Получить практические навыки создания абстрактных типов данных и перегрузки

операций.

**Основное содержание работы.**

Определить и реализовать класс − абстрактный тип данных. Определить и реализовать операции

над данными этого класса.

**Порядок выполнения работы.**

1. Выбрать класс АТД в соответствии с вариантом.
2. Определить и реализовать в классе конструкторы, деструктор,

функции Input (ввод с клавиатуры) и Print (вывод на экран), перегрузить операцию присваивания.

1. Написать программу тестирования класса и выполнить тестирование.
2. Дополнить определение класса заданными перегруженными операциями ( в
3. соответствии с вариантом).
4. Реализовать эти операции. Выполнить тестирование.
5. Реализовать вариант на языке С++ и (С#/Java на выбор).

Для моего варианта предложены следующие операции:

АТД − множество с элементами типа char. Дополнительно перегрузить следующие

операции:

- − удалить элемент из множества (типа set-char);

> − проверка на подмножество;

!= − проверка множеств на неравенство.

**Задание 1. Определяем и реализовываем в классе конструкторы, деструктор, перегрузка операции:**

1. //Конструктор
2. SetChar() {
3. setChar = NULL;
4. this->size = 0;
5. cout << "Kontstryktor" << endl;
6. }
7. // добавл в конструктор
8. void addSymbol(char sym) {
9. char\* tempSet = new char[size + 1];
10. for (int i = 0; i < size; i++) {
11. tempSet[i] = this->setChar[i];
12. }
13. tempSet[size] = sym;
14. // delete old char set
15. delete[] this->setChar;
16. // reasing
17. this->setChar = tempSet;
18. size++;
19. }
20. // конструктор с параметрами
21. SetChar(char\* charArray, unsigned int size) {
22. this->setChar = new char[size];
23. for (int i = 0; i < size; i++) {
24. addSymbol(charArray[i]);
25. }
26. }
27. // Конструктор копирование
28. SetChar(const SetChar& ob) {
29. size = ob.size;
30. setChar = new char[ob.size];
31. for (int index = 0; index < size; index++)
32. setChar[index] = ob.setChar[index];
33. }
34. //Деструктор
35. ~SetChar() {
36. if (setChar != NULL) {
37. delete[] setChar;
38. }
39. }
40. // перегрузка присваивания
41. SetChar& operator = (const SetChar& ob) {
42. if (size > ob.size) {
43. for (int index = 0; index < ob.size; index++)
44. setChar[index] = ob.setChar[index];
45. for (int index = ob.size; index < size; index++)
46. setChar[index] = 0;
47. }
48. else {
49. for (int index = ob.size; index < size; index++)
50. setChar[index] = ob.setChar[index];
51. }
52. return \*this;
53. }
54. //вывод значений оператора перегрузок
55. friend ostream& operator<<(ostream& output, const SetChar& r)
56. {
57. if (r.charSet != NULL) {
58. for (int i = 0; i < r.currentSize; i++) {
59. output << r.charSet[i] << ' ';
60. }
61. }
62. else {
63. output << "empty";
64. }
65. return output;
66. }

**Задание 2. Дополнить определение класса заданными перегруженными операциями:**

1. SetChar& operator-(char del) { //удаление
2. for (int index1 = 0; index1 < size; index1++) {
3. if (setChar[index1] == del)
4. for (int index2 = index1; index2 < size + 1; index2++) {
5. setChar[index2] = setChar[index2 + 1];
6. }
7. }
8. size -= 1;
9. return \*this;
10. }
11. bool operator!=(SetChar ob) { //на неравенство
12. for (int i = 0; i < this->size; i++) {
13. if (ob.find(this->setChar[i]) == -1) {
14. return false;
15. }
16. }
17. return true;
18. }
19. friend bool operator>(const SetChar& first, const SetChar&
20. second) { //проверка множество на подмножество
21. if (first.size > second.size) {
22. int count = 0;
23. if (first.setChar != NULL && second.setChar != NULL) {
24. for (int i = 0; i < first.setChar[i]; i++) {
25. for (int j = 0; j < second.setChar[j]; j++) {
26. if (first.setChar[i] == second.setChar[j]) {
27. count++;
28. }
29. }
30. }
31. }
32. else {
33. return true;
34. }
35. if (count == second.size) {
36. return true;
37. }
38. else {
39. return false;
40. }
41. }
42. else {
43. return false;
44. } }

**Задание 3. Выполнить тестирование класса с перегрузками:**

1. int main(void) {
2. char xVector[] = { '1', '2', '3', '1' };
3. char yVector[] = { '4', '2' };
4. SetChar xSetChar(xVector, 4), ySetChar;
5. cout << "x SetChar: " << xSetChar << endl;
6. cout << "y SetChar: " << ySetChar << endl;
7. for (int i = 0; i < 2; i++) {
8. ySetChar = ySetChar - yVector[i];
9. }
10. xSetChar = xSetChar;
11. xSetChar = xSetChar;
12. xSetChar = xSetChar;
13. cout << "x SetChar: " << xSetChar << endl;
14. cout << "y SetChar: " << ySetChar << endl;
15. SetChar zSetChar;
16. zSetChar = xSetChar - '1';
17. zSetChar = zSetChar - 'a';
18. zSetChar = zSetChar - 'x';
19. zSetChar = zSetChar - xSetChar;
20. zSetChar = zSetChar - ySetChar;
21. zSetChar = zSetChar - zSetChar;
22. cout << "z SetChar: " << zSetChar << endl;
23. if (xSetChar != xSetChar) {
24. cout << "xSetChar equels xSetChar" << endl;
25. }
26. else {
27. cout << "Error in == operator" << endl;
28. }
29. if (!(ySetChar != zSetChar)) {
30. cout << "ySetChar not equels zSetChar" << endl;
31. }
32. else {
33. cout << "Error in == operator" << endl;
34. }
35. return 0;
36. }