**Ръководство за програмиста**

* **Структура PC**:
  + *поле id* – целочислен тип, уникално за всеки обект
  + *полета date[], warranty[], model[], hard\_drive[]* – масиви от символи (низове)
  + *поле ram* – реален тип (измерено в гигабайти)
  + *поле price* – реален тип (измерено в лева)
  + *void Input()* – функция за въвеждане на полетата
  + *void Output()* – функция за извеждане на информацията на обекта
* **Глобални променливи**:
  + *number\_pcs* – от целочислен тип, представлява броя на въведените компютри
  + *pcs[]* – масив от всички въведени компютри (тип PC)
* **int EnterId()** – използва се за въвеждане на поле id във void Input() на PC, като проверява за неговата уникалност, връща променлива от целочислен тип
* **bool IsInArr(int arr[], int sizeArr, int x)** – използва се в SortStr() и SortNum(), приема аргументи arr[] – масив от цели числа, sizeArr – цяло число, големината на arr[] и x – цяло число; функцията проверява дали x присъства в масива arr[] и връща променлива от булев тип
* **void Input()** – опция 1, въвежда определен брой компютри и тяхната информация в масива pcs[]
* **void Output()** – опция 2, извежда всички компютри и тяхната информация от масива pcs[]
* **void Virus()** – опция 3, изпълва екрана със случайни символи, безкраен цикъл
* **void SearchNum()** – опция 4, търси по въведено от конзолата реално число за съвпадение с числовите полета на всички компютри от масива pcs[] и извежда резултата
* **void SearchId()** – опция 5, търси по въведено цяло число за съвпадение с полетата id на компютрите в pcs[] и извежда резултата
* **void SearchStr()** – опция 6, търси по въведен низ за съвпадение с низовите полета на компютрите от pcs[] и извежда резултата
* **void Compare()** – опция 7, извежда броя на компютрите в pcs[], чието избрано числово поле има по-голяма стойност от въведено реално число
* **void Sum()** – опция 8, извежда сумите на полетата ram и price на всички компютри от масива pcs[]
* **int CompareStr(char str1[], char str2[], int i = 0)** – сравнява 2 низа по азбучен ред чрез рекурсия, използва се в SortStr(), приема аргументи str1[] и str2[] – 2-та сравнявани низа и i – цяло число, което съхранява позицията на сравняваната буква; функцията извежда цяло число: 0, ако двата низа са еднакви, -1, ако str1 е преди str2 по азбучен ред и 1, ако str1 е след str2 по азбучен ред
* **void SortStr()** – опция 9, извежда компютрите от pcs[], сортирани по азбучен ред спрямо избрано от потребителя низово поле
* **void SortNum()** – опция 10, извежда компютрите от pcs[], сортирани по нарастващ ред спрямо избрано от потребителя числово поле
* **void Delete()** – опция 12, изтрива компютър от pcs[] по дадена от потребителя позиция (целочислен тип) и премества последния елемент на pcs[] на позицията на изтрития
* **void Change()** – опция 13, променя стойността на избрано поле от избран компютър от pcs[]
* **void Menu()** – рекурсивна функция, извежда менюто за избор на потребителя, чрез switch statement избира по въведено от потребителя цяло число от диапазона [0; 12] коя функция да се изпълни:
  + *опция 0* – излиза от програмата
  + *опция 1* – Input()
  + *опция 2* – Output()
  + *опция 3* – Virus()
  + *опция 4* – SearchNum()
  + *опция 5* – SearchId()
  + *опция 6* – SearchStr()
  + *опция 7 –* Compare()
  + *опция 8 –* Sum()
  + *опция 9 –* SortStr()
  + *опция 10 –* SortNum()
  + *опция 11 –*въвежда нов компютър и неговата информация като последен елемент на pcs[]
  + *опция 12 –* Delete()
  + *опция 13 –* Change()
* **int Main() –** изпълнява се при стартиране на програмата, извиква Menu(), връща цяло число 0.