**Факултет „Автоматика“**

**Курсов проект по ПИК-2**

**Изготвил:**

**Михаил Узунов**

**Факултет Автоматика, група 1 ф No. 011215064**

***Дата: 06.04.2016г. Проверил: ................***

**Съдържание**

1. **Задание..........................................................................................................**
2. **Описание на функциите...............................................................................**

**Задание**

1. Да се състави програма, която да поддържа масив от цели числа до 100 елемента.
2. Програмата да има текстово меню, от което потребителят да може да избира следните операции:
   * Попълване на масива (потребителя предварително да избере с колко числа ще работи)
   * Проверка за наличие на елемент по стойност.
   * Отпечатване на съдържанието на масива на екрана
   * Допълнителни условия в зависимост от варианта.
   * Зареждане на масив от посочен по време на работа на програмата файл.
   * Запис на съдържанието на масива в посочен по време на работа на програмата файл.
   * Изход.

Вариант 6: Да се намери елемента от масива, който е най-близо по абсолютна стойност до средно аритметичното на числата от масива.

**Описание на функциите**

***int main()***

**Входни данни:**

**Няма.**

**Изходни данни:**

**Връща тип integer, резултат на изпълнение на програмата.**

**Декларирани променливи във main():**

* **option (*char*) - Променлива в която се съхранява избора на потребителя.**
* **should\_run (*int*) - Флаг за маркиране за изход от програмата.**
* **elements (*int*) - Променлива в която се съхранява големината на масива**
* **arr (int\*) - Указател към първия елемент на масива**

**Структура на функцията main():**

**Създава се цикъл с флаг should\_run. Принтира се менюто на програмата и се иска от потребителя да въведе операцията която иска да извърши програмата.**

***void init(int\*\* p, int size)***

**Входни данни:**

* **p (int\*\*) – Указател към указателя, за да може да се използва във функция. Използва се за да съхранява указателя на масива подаден от main()**
* **size (int) – Променливата съхранява големината на масива**

**Изходни данни:**

**Няма.**

**Декларирани променливи във init(int\*\* p, int size):**

* **max\_size (*int*) – максималната големина на масива**

**Структура на функцията:**

**Проверява се първо дали подадената големина на масива не е по-голяма от 100 и след това се заделя памет, чрез функцията calloc(), като и се подава колко елемента трябва да задели (size) и големината на тип (int). Тя връща указател, който се присвоява на указателя p. Ако големината на size е по-голяма от 100 се принтира съобщение.**

***void print\_menu()***

**Входни данни:**

**Няма.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

**Няма.**

**Структура на функцията:**

**Принтира се следния текст:**

1. **Fill array with numbers**
2. **Find number in array**
3. **Print array**
4. **Find element closest to the average sum**
5. **Load array from file**
6. **Save array to file**
7. **Exit**

***void cons\_clear()***

**Входни данни:**

**Няма.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

**Няма.**

**Структура на функцията:**

**Извиква се функцията system(“cls”), която изчиства конзолата.**

***void n\_printf(char\* str)***

**Входни данни:**

* **str (*char\**) – Съхранява стринг, който ще се извика в printf();**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

**Няма.**

**Структура на функцията:**

**Извиква се функцията printf(), като й се подава str и накрая добавя нов ред (\n).**

***void fill\_array(int\* p, int size)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

* **max\_size (*int*) – максималната големина на масива**

**Структура на функцията:**

**Проверява се, дали size е по-малко от 100, и се пуска for цикъл да попълни масива.**

***void print\_array(int\* p, int size)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

* **max\_size (*int*) – максималната големина на масива**

**Структура на функцията:**

**Проверява се, дали size е по-малко от 100, и се принтира всеки елемент от масива.**

***void flush\_array(int\* p, int size)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

**Няма.**

**Структура на функцията:**

**Функцията пуска for цикъл, който занулява всеки елемент в масива.**

***void average(int\* p, int size)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**

**Изходни данни:**

**Връща тип float - средно аритметично.**

**Декларирани променливи във функцията:**

* **sum (*int*) – сумата на събраните елементи**
* **count (*int*) - брой намерени числа**

**Структура на функцията:**

**Минава се със for цикъл през масива и се събират всички елементи. След това разделя сумата на намерените елементи и връща това число.**

***void find\_closest(int\* p, int size, float avg)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**
* **avg (float) – Средно аритметично на масива**

**Изходни данни:**

**Връща тип int със числото най-близо до средно аритметичното.**

**Декларирани променливи във функцията:**

* **closest (*int*) – максималната големина на масива**

**Структура на функцията:**

**Минава се през целия масив като се взима абсолютната стойност на сегашния елемент, изваждайки от него средно аритметичното и се сравнява с последния елемент най –близък до средно аритметичното, изваждайки средното аритметично и ако е по-малко се записва в променливата closest.**

***void save\_array(int\* p, int size, char\* fn)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**
* **fn (*char\**) – Съхранява името на файла.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

* **fp (*File\**) – Указател за файла в който ще се записват стойностите от масива.**

**Структура на функцията:**

**Първо отваряме файла с функцията fopen() като и се подаде името на файла fn и опция “w” (write). Проверяваме дали файла е валиден, и ако е записваме в него елементите от масива със функцията fprintf().**

***void save\_array(int\* p, int size, char\* fn)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**
* **fn (*char\**) – Съхранява името на файла.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

* **fp (*File\**) – Указател за файла в който ще се записват стойностите от масива.**

**Структура на функцията:**

**Първо отваряме файла с функцията fopen() като и се подаде името на файла fn и опция “r” (read). Проверяваме дали файла е валиден, и ако е четем от него елементите от масива със функцията fscanf().**

***void err\_empty()***

**Входни данни:**

**Няма.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

**Няма.**

**Структура на функцията:**

**Функцията принтира съобщение, че масивът е празен!**

***void find\_number(int\* p, int size, int target)***

**Входни данни:**

* **p (*int\**) – Указател към първия елемент на масив**
* **size (*int*) – Големината на масива**
* **target (*int*) – Числото което трябва да се намери.**

**Изходни данни:**

**Няма.**

**Декларирани променливи във функцията:**

* **bFound (*int*) – Флаг дали е намерено числото в масива**

**Структура на функцията:**

**Минава се през целия масив и се проверява дали текущия елемент е равен на числото което се търси. Ако се намери, се спира цикъла и се вдига флага. Накрая се принтира съобщение дали е намерено числото, или не е.**