Realizati un program in C care tine evidenta uneltelor dintr-un magazin de unelte.

Fiecare unealta este descrisa de: nume, marcă, cod, pret.

Realizati un meniu interactiv cu urmatoarele optiuni:

0. Iesire.

1. Adaugarea unei unelte.

2. Afisarea uneltelor din magazin.

3. Cautarea unei unelte după nume, numele este dat de utilizator de la tastatura.

4. Sa se afiseze toate uneltele care incep cu litera ‘c’.

5. Sa se afiseze uneltele care au un pret mai mic decât un pret dat de utilizator de la

tastatura.

6. Sa se copieze într-un tablou pretul fiecarei unelte, apoi sa se sorteze acel tablou in

ordine descrescatoare, iar la final sa se afiseze tabloul.

7. Sa se afiseze reprezentarile binare ale pretului – se vor utiliza operatii pe biti pentru

reprezentarea binara.

8. Sa se afiseze cate unelte au pretul mai mic cu 1 decat o putere a lui 2 – se vor

utiliza operatii pe biti pentru aflarea daca pretul este mai mic cu 1 decat o putere a lui

2.

9. Sa se afiseze pe diagonala secundara a unei matrice, de ordin dat de numarul de

unelte, lungimea cea mai mare a marcii uneltelor, iar pe restul elementelor sa se puna

0.

10. Sa se afiseze care este unealta cu pretul cel mai mare.

Observatii:

- Numele uneltei este considerat citit corect, daca este format strict din litere mici

sau cifre.

- Marca este considerata citita corect daca are cel putin 2 caractere.

- Codul este dat automat fiint egal cu ultimele doua caractere din marca.

- Pretul este considerat citit corect daca este mai mare decat 5.

- Pentru fiecare funcționalitate se va creea o funcție.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

typedef struct

{

char nume[20];

char marca[20];

char cod[3];

int pret;

}Unelte;

int validare\_nume(char nume[])

{

for (int i = 0; i < strlen(nume); i++)

{

if (!islower(nume[i]) && !isdigit(nume[i]))

return 0;

}

return 1;

}

int validare\_marca(char marca[])

{

if (strlen(marca) < 2)

return 0;

return 1;

}

int validare\_pret(int pret)

{

if (pret <= 5)

return 0;

return 1;

}

void adaugare(Unelte unelte[], int\* n)

{

Unelte u;

int x;

printf("Introduceti numele uneltei: ");

scanf("%s", u.nume);

if (!validare\_nume(u.nume))

{

printf("Mai incearca!");

return;

}

printf("Inroduceti marca: ");

scanf("%s", u.marca);

if (!validare\_marca(u.marca))

{

printf("Mai incearca!");

return;

}

printf("Introduceti pretul:");

scanf("%d", &u.pret);

if (!validare\_pret(u.pret))

{

printf("Mai incearca!");

return;

}

x = strlen(u.marca);

strncpy(u.cod, &u.marca[x-2], 2);

u.cod[2] = '\0';

unelte[\*n] = u;

(\*n)++;

printf("Unealta a fost adaugata!");

}

void afisare(Unelte unelte[], int n)

{

printf("Uneltele din magazin sunt:\n");

printf("%-20s %-20s %-10s %s\n", "Nume", "Marca", "Cod", "Pret");

for (int i = 0; i < n; i++)

printf("%-20s %-20s %-10s %d\n", unelte[i].nume, unelte[i].marca, unelte[i].cod, unelte[i].pret);

}

void cautare(Unelte unelte[], int n)

{

char nume\_cautat[50];

printf("Introduceti numele cautat:");

scanf("%s", nume\_cautat);

int gasit = 0;

for (int i = 0; i < n; i++)

if (strcmp(unelte[i].nume, nume\_cautat) == 0)

{

printf("Uneltea a fost gasita: \n");

printf("Nume: %s\n", unelte[i].nume);

printf("Marca: %s\n", unelte[i].marca);

printf("Cod: %s\n", unelte[i].cod);

printf("Pret: %d\n", unelte[i].pret);

gasit = 1;

break;

}

if (gasit == 0)

printf("Nu a fost gasita nici o unealta cu acest nume!");

}

void afisare\_incepe\_cu\_c(Unelte unelte[], int n)

{

printf("Uneltele care incep cu 'c' sunt: \n");

for (int i = 0; i <= n; i++)

if (unelte[i].nume[0] == 'c')

printf("%s\n", unelte[i].nume);

}

void afisare\_pret\_mai\_mic(Unelte unelte[], int n)

{

int pret\_max;

printf("Introduceti pretul maxim");

scanf("%d", &pret\_max);

printf("Unelete a caror pret e mai mic decat %d: \n", pret\_max);

for (int i = 0; i <= n; i++)

if (unelte[i].pret < pret\_max)

{

printf("Nume: %s\n", unelte[i].nume);

printf("Marca: %s\n", unelte[i].marca);

printf("Cod: %s\n", unelte[i].cod);

printf("Pret: %d\n", unelte[i].pret);

}

}

void tablou\_preturi(Unelte unelte[], int n)

{

int preturi[50];

for (int i = 0; i <= n; i++)

preturi[i] = unelte[i].pret;

for (int i = 0; i < n - 1; i++)

for (int j = i + 1; j < n; j++)

if (preturi[i] < preturi[j])

{

int temp = preturi[i];

preturi[i] = preturi[j];

preturi[j] = temp;

}

printf("Preturile uneltelor sortate descrescator:\n");

for (int i = 0; i < n; i++)

printf("%d ", preturi[i]);

}

void afisare\_reprezentare\_binar(Unelte unelte[], int n)

{

int preturi[50];

for (int j = 0; j <= n; j++)

preturi[j] = unelte[j].pret;

for (int j = 0; j < n; j++)

{

printf("Pretul uneltei %s in zecimal este %d, iar in binar :", unelte[j].nume, unelte[j].pret);

for (int i = 31; i >= 0; i--)

{

printf("%d", (preturi[j] >> i) & 1);

}

printf("\n");

}

printf("\n");

}

void afiseaza\_preturile\_mai\_mici\_cu\_1\_put\_2(Unelte unelte[], int n)

{

for (int i = 0; i <= n; i++)

if ((unelte[i].pret & (unelte[i].pret + 1)) == 0)

printf("Unealta '%s' are pretul: %d, care este cu 1 mai mic decat o putere a lui 2 \n", unelte[i].nume, unelte[i].pret);

printf("\n");

}

void matrice(Unelte unelte[], int n)

{

int M[10][10];

int lungime\_maxima\_marca = 0;

for (int i = 0; i < n; i++)

{

int lungime\_marca = strlen(unelte[i].marca);

if (lungime\_marca > lungime\_maxima\_marca)

lungime\_maxima\_marca = lungime\_marca;

}

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

if (i+j == n- 1)

M[i][j] = lungime\_maxima\_marca;

else

M[i][j] = 0;

}

}

printf("Matricea ceruta este urmatoarea:\n");

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

printf("%d ", M[i][j]);

printf("\n");

}

printf("\n");

}

void afisare\_pret\_maxim(Unelte unelte[], int n)

{

int pret\_max=0;

int preturi[50];

int index\_max;

for (int i = 0; i <= n; i++)

preturi[i] = unelte[i].pret;

for (int i = 0; i < n; i++)

if (preturi[i] < preturi[i + 1])

{

index\_max = i + 1;

pret\_max = preturi[i + 1];

}

printf("Unealta cu pretul cel mai mare este: %s, are pretul de %d", unelte[index\_max].nume, pret\_max);

printf("\n");

}

int main()

{

Unelte unelte[50];

int n = 0;

int optiune;

do {

printf("0. Iesire\n");

printf("1. Adauga unealta\n");

printf("2. Afisare unealta \n");

printf("3. Cautare\n");

printf("4. Afisare unelte care incep cu 'c'\n");

printf("5. Afisare unelte cu pret mai mic decat cel introdus\n");

printf("6. Afisare descrescator tablou preturi\n");

printf("7. Afisare preturi in binar\n");

printf("8. Afisare preturi cu 1 mai mic decat o putere a lui 2\n");

printf("9. Afisare matrice cu diagonala secundara = cea mai lunga marca, in rest = 0\n");

printf("10. Afisarea uneltei cu cel mai mare pret\n");

printf("Optiunea aleasa:");

scanf("%d", &optiune);

switch (optiune)

{

case 1:

adaugare(unelte, &n);

break;

case 2:

afisare(unelte, n);

break;

case 3:

cautare(unelte, n);

break;

case 4:

afisare\_incepe\_cu\_c(unelte, n);

break;

case 5:

afisare\_pret\_mai\_mic(unelte, n);

break;

case 6:

tablou\_preturi(unelte, n);

break;

case 7:

afisare\_reprezentare\_binar(unelte, n);

break;

case 8:

afiseaza\_preturile\_mai\_mici\_cu\_1\_put\_2(unelte, n);

break;

case 9:

matrice(unelte, n);

break;

case 10:

afisare\_pret\_maxim(unelte, n);

break;

default:

printf("Nu exista asa optiune!");

break;

}

printf("\n");

} while (optiune != 0);

}