ԿՐԹՈՒԹՅԱՆ ԵՎ ԳԻՏՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

ՀԱՅԱՍՏԱՆԻ ԱԶԳԱՅԻՆ ՊՈԼԻՏԵԽՆԻԿԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ

ԿՈՒՐՍԱՅԻՆ ԱՇԽԱՏԱՆՔ

ԻՆՖՈՐՄԱՏԻԿԱ առարկայից

Ամբիոն՝ ՏԱԾԱ

Խումբ՝ SS055-1

Ուսանող՝ Օրդուխանյան Էդուարդ _____
ստորագրություն

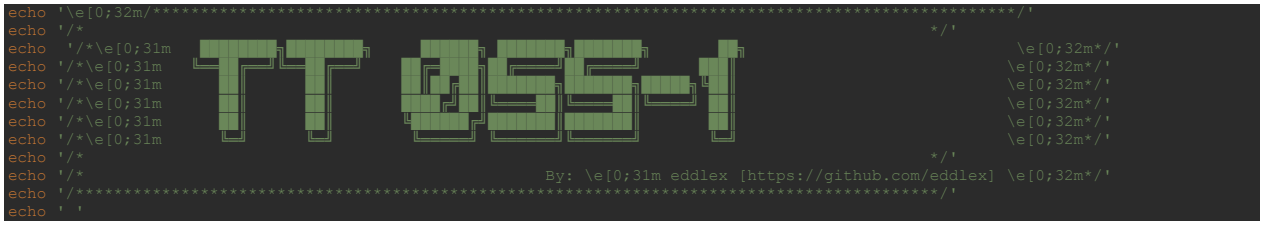
Դասախոս՝ Վ.Ղ.Ղուկասյան

ԱՌԱՋԱԴՐԱՆՔ 22

Տրված N×M չափողականու թյամբ ուղղանկյուն մատրիցը, որի տարրերը ամբ ողջ թվեր են, արտածել տեսատիպի էկր անի վրա: Կազմել ծրագիր, որը կձևավորի նոր մատրից, տեղափոխելով այնտեղ տրվա մատրիցի այն տողերը, որոնք չեն կարգավորված ըստ արժեքների աճման կամ նվազման: Նոր մատրիցի ձևավորումը ցուցադրել դանդաղեցված ռեժիմում՝ առաջին մատրիցի կողքին: Տեղափոխվող տողերը տրված մատրիցից հեռացնել:

ԱԼԳՈՐԻԹՄԻ ՆԿԱՐԱԳՐՈՒԹՅՈՒՆ

[header.sh]



[libft.h]

```
#ifndef LIBFT_H

# define LIBFT_H

# include <stdio.h>   // —> printf()

# include <stdlib.h>  // —> malloc()

# include <unistd.h>  // —> sleep() || usleep()

typedef struct cube         //}

{                           //} —> Հայտարարում է MX տիպի struct որը պարունակում է՝

  int n;                    //} —> n տողերի քանակ,

   int m;                   //} —> m սյուների քանակ,

   int   **arr;             //} —> arr ցուցիչի ցուցիչ:

} MX;                       //}

void   sh();                        //—> Տպում է header.sh file_ի պարունակությունը և գույնը reset է անում:

void   color (int flag);            //—> Փոխում է console_ի գույնը ըստ flag_ի:

void   print(MX *mat);              //—> Տպում է matrix_ը:
```

```
void   info();                          //—> Տպում է խնդրի պահանջը:
void   ex22();                          //—> 22 համարի վարժության ֆունկցիա:
void   ft_free(MX *mat);                //—> Մաքրում է դինամիկ հիշողությունը:
void   change(MX *mat);                 //—> Անիմացիայի համար պատասխանատու ֆունկցիա:
void   create_arr(MX *mat);             //—> Ստեղծում է matrix:
void   push_random(MX *mat);            //—> Ռանդոմ տարբերակով էլեմենտներ է ավելացնում matrix_ի մեջ:
void   N_and_M(MX *mat);                //—> Ստուգում է N_ի և M_ի ճիշտ լինելու պայմանը  [N > 1; M > 2];
int    not_sorted(MX *mat, int n);      //—> Եթե տողը սորտավորված է վերադարցնում է 1 հակառակ դեպքում 0:
#endif
```

[print.c]

```
#include "libft.h"
void sh()
{
   system("sh header.sh");
   color(0);
}
void info()
{
   sh();
   color(1);
   printf("Տրված N×M չափողականու թյամբ ուղղանկյուն մատրիցը, որի տարրերը ամբ ողջ թվեր են, արտածել\n");
   printf("տեսատիպի էկր անի վրա: Կազմել ծրագիր, որը կձևավորի նոր մատրից, տեղափոխելով այնտեղ տրված\n");
   printf("մատրիցի այն տողերը, որոնք չեն կարգավորված ըստ արժեքների աճման կամ նվազման: Նոր մատրիցի\n");
   printf("ձևավորումը ցուցադրել դանդաղեցված ռեժիմում՝ առաջին մատրիցի կողքին: Տեղափոխվող տողերը\n");
   printf("տրված մատրիցից հեռացնել:\n\n");
   color(2);
   printf("press any button ...");
   getchar();
   system("clear"); // -> ջնջում է console_ի պարունակությունը:
}
```

```
void color (int flag)
{
  if (flag == 0)
    printf("\033[0m");    //reset
  else if (flag == 2)
    printf("\033[1;31m");  //red
  else if (flag == 1)
    printf("\033[0;32m");
  else if (flag == 3)      //green
    printf("\033[0;32m");
}


void print(MX *mat)
{
   system("clear");
   sh();
   for (int i =  0; i < mat->n; ++i)
   {
      if (not_sorted(mat, i))
         color(2);
      else
         color(3);
      for (int j = 0; j < mat->m * 2; ++j)
      {
         if ( j >= mat->m)
            color(2);
       if(j == mat->m)
         printf("   ");
       if( mat->arr[i][j])
         printf("%d ", mat->arr[i][j]);
       else
         printf("  ");
      }
      printf("\n");
      color(1);
   }
   usleep(170000); //-> Դանդաղեցնում է code_ի աշխատանքը:
}
```

[ex22.c]

```
#include "libft.h"

int not_sorted(MX * mat, int n)

{

   int min_max = 1;

   int max_min =  1;

   for (int i = 0; i < mat->m - 1; ++i)

   {

      if (mat->arr[n][i] < mat->arr[n][i + 1])

         max_min = 0;

      if (mat->arr[n][i] > mat->arr[n][i + 1])

         min_max = 0;

      if (!mat->arr[n][i])

         return (1);

   }

   if (max_min && !min_max || min_max && !max_min)

      return(0);

   return (1);

}


void N_and_M(MX * mat)

{

   system("sh header.sh");

   char str[20] = "-1";

   mat->n = -1;

   mat->m = -1;

   while (mat->n <= 0)

   {

      printf("input number N: ");

      color(2);

      scanf("%s", str);

      mat->n = atoi(str);

      if (mat->n <= 0)

         printf("error N\n");

      color(1);

   }

   while (mat->m <= 0)
```

```
    {
        printf("input number M: ");
        color(2);
        scanf("%s", str);
        mat->m = atoi(str);
        if (mat->m <= 0)
            printf("error M\n");
        color(1);
    }
    sleep(1);
    system("clear");
}
void create_arr(MX *mat)
{
    if (!(mat->arr = (int**)malloc(sizeof(int*) * mat->n)))
        return;
    for (int i = 0; i < mat->n; ++i)
        if (!(mat->arr[i] = (int*)malloc( sizeof(int) *  (mat->m * 2) )))
            return;
}
void push_random(MX * mat)
{
    for(int i = 0; i < mat->n; ++i)
        for (int j = 0; j < mat->m; ++j)
            mat->arr[i][j] = rand() % 8  + 1;
}
void change(MX * mat)
{
    for (int i = 0; i < mat->n; ++i)
    {
        int count;
        if (not_sorted(mat, i))
        {
            for (int c = mat->m -1 ; c >= 0; --c)
            {
                count = c;
                for (int k = 0; k < mat->m; ++k)
                {
```

```
                mat->arr[i][count + 1] = mat->arr[i][count];
                mat->arr[i][count] = 0;
                print(mat);
                ++count;
            }
            int ii = i;
            while (ii != 0 && mat->arr[ii - 1][count] == 0)
            {
                mat->arr[ii - 1][count] = mat->arr[ii][count];
                mat->arr[ii][count] = 0;
                print(mat);
                --ii;
            }
        }
    }
    else
    {
        for (int x = 0; x <= mat->m; ++x)
        {
            int ii = i;
            while (ii != 0 && mat->arr[ii - 1][x] == 0)
            {
                mat->arr[ii - 1][x] = mat->arr[ii][x];
                mat->arr[ii][x] = 0;
                print(mat);
                --ii;
            }
        }
    }
  }
}
void ft_free(MX * mat)
{
  for (int i = 0; i < mat->n; ++i)
    free(mat->arr[i]);
  free(mat->arr);
  free(mat);
}
```

```
void ex22()
{
   info();
   MX *mat = (MX*)malloc(sizeof(MX));
   N_and_M(mat);
   create_arr(mat);
   push_random(mat);
   print(mat);
   sleep(1);
   change(mat);
   ft_free(mat);
}
```

[main.c]

```
#include "libft.h"
int main (void)
{
  ex22();
  return (0);
}
```

Git: eddlex/polytech_22 (github.com)

RUN gcc main.c print.c ex22.c && ./a.out