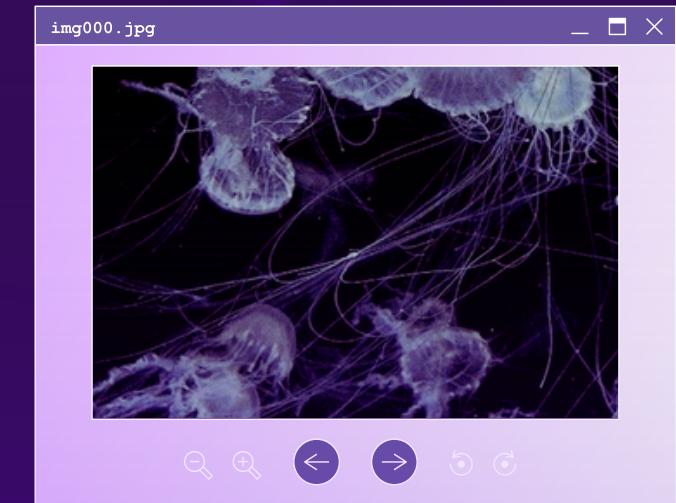
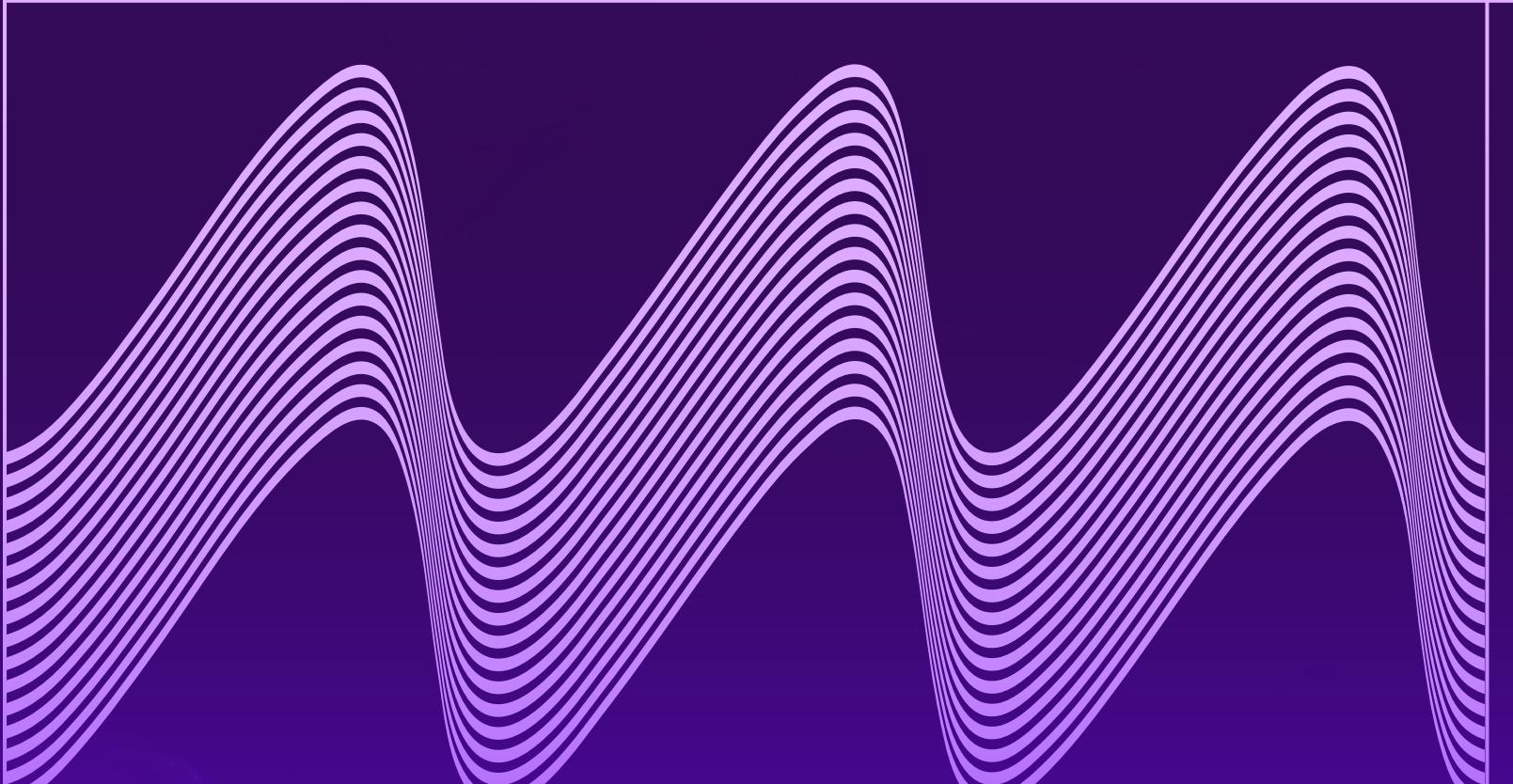
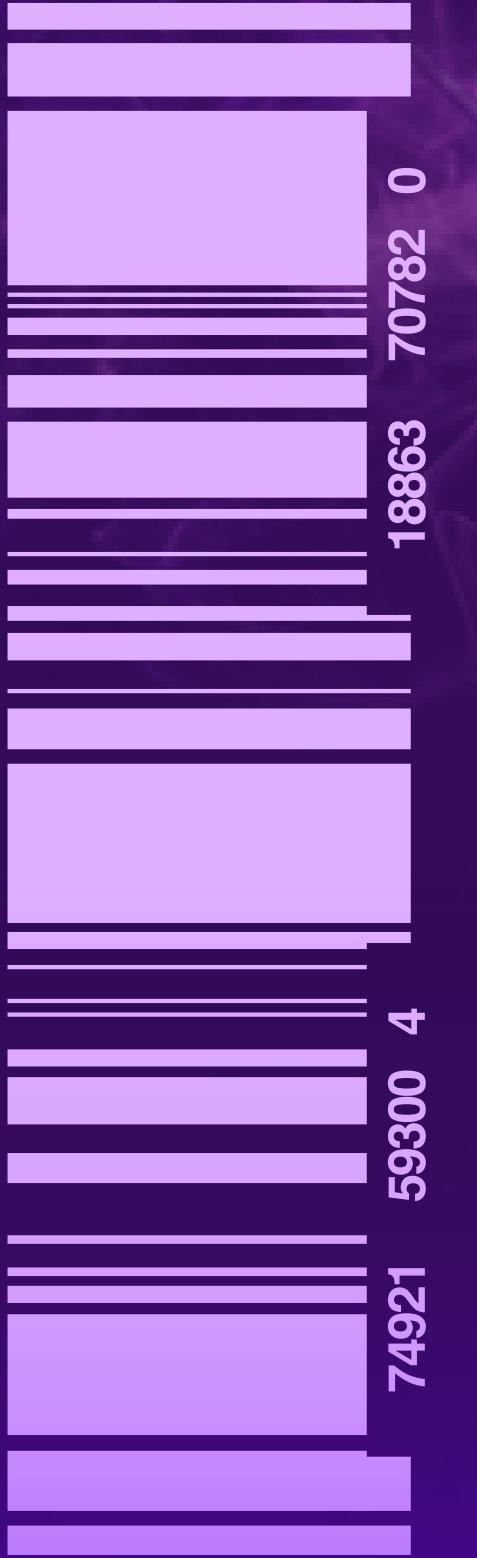


ПРОЕКТ

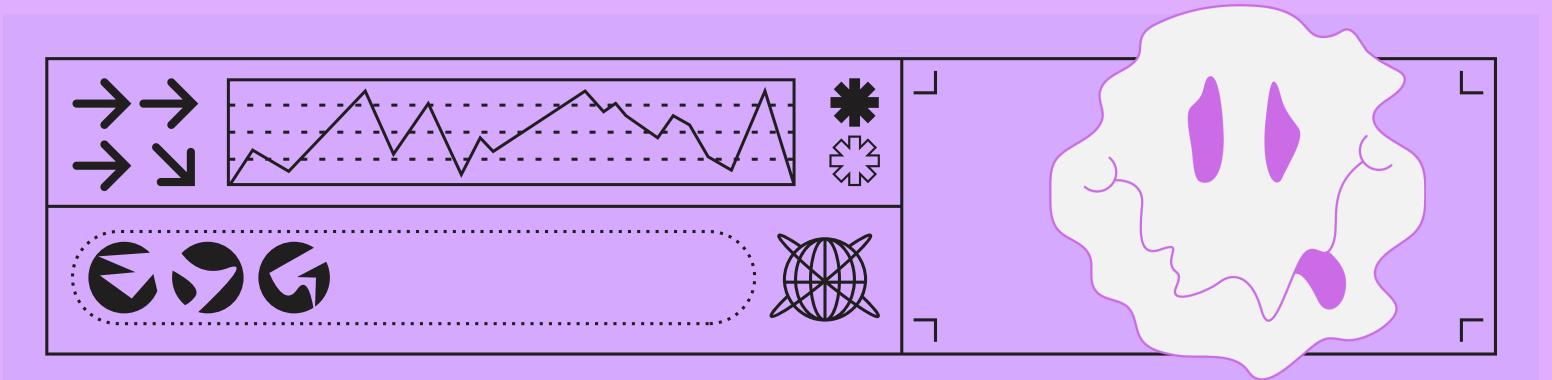
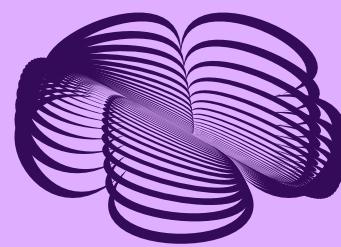
PRIORI-TIME-MANAGEMENT



Anastasiia
Nachynka
PMk-12s



ПЛАН?



↓ ПРО ЩО ПОГОВОРИМО ↓

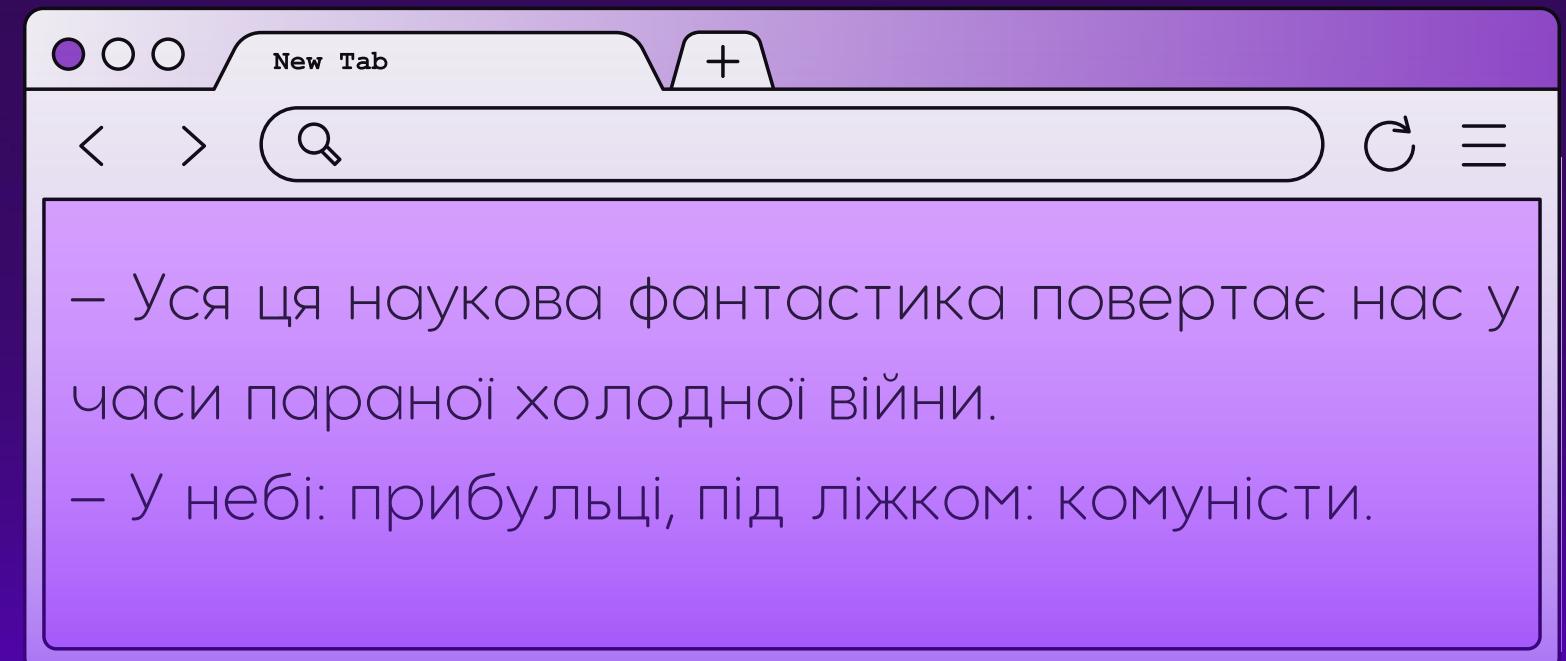
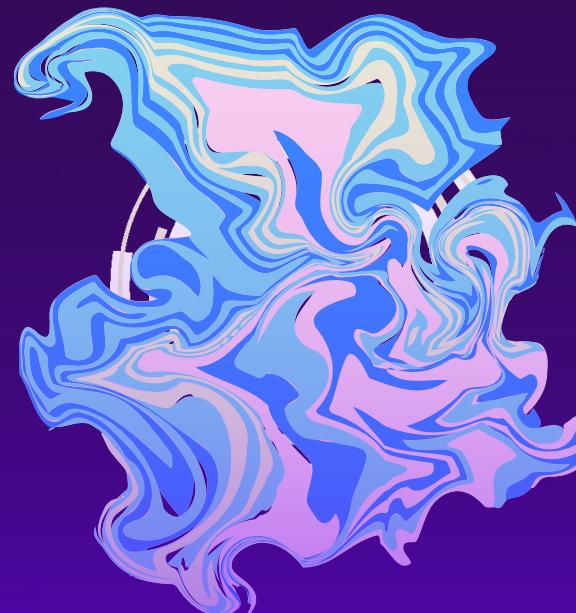
Для чого?

Як працює?

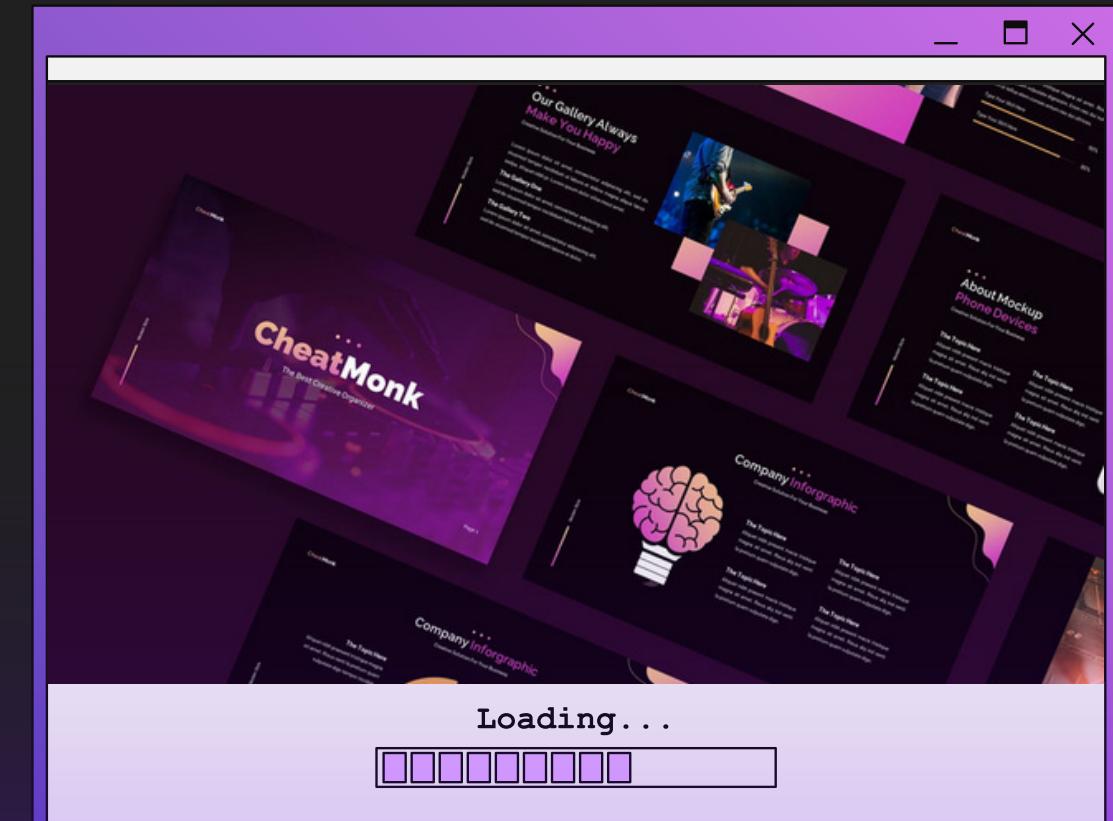
Чи є аналоги?

Яко конструкція?

Як виглядає?

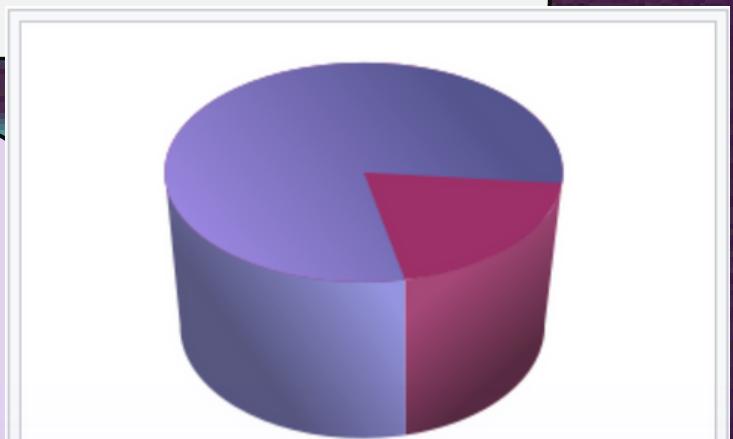


ДЛЯ ЧОГО БУЛО СТВОРЕНО PRIORI- TІME-MANAGEMENT



Керування часом, тайм-менеджмент (від англ. time management) – сукупність методик оптимальної організації часу для виконання поточних задач, проектів та календарних подій. Типовими підходами в керуванні часом є постановка пріоритетів, розбиття великих завдань та проектів на окремі дії та делегування іншим людям.

Тайм-менеджмент (планування часу) – це технології організації та оптимізації часу. Вони допоможуть, якщо ви спізнюютеся, не встигаєте виконати все задумане або постійно відкладаєте важливі справи..



Згідно з [принципом Парето](#), лише 20 % всієї діяльності призводить до 80 % бажаного результату.



ЯК ЦЕ ПРАЦЮЄ?



Уявімо, що вам дуже складно займатися самоорганізацією, але у вас є таски, і їх треба виконувати (до завершення дедлайну). Час іде, а ви все ще не почали займатися потрібними справами. Чим більше часу проходить – тим більше "завдань" накопичується, і поволі збільшується шанс просто прогавити дедлайн.

Тоді вам точно у нагоді стануть "нагадувалки".

Тайм-менеджмент (планування часу) – своєрідний органайзер для планування вашого часу, де ви зможете внести певний таск, обрати дату (термін) для виконання, а також позначити завдання як "важливе чи не дуже", що допоможе вам лінійно організувати власний час та виконати все не прогавиви "дедлайни".

74921 59300 4 18863 70782 0



```
#include <iostream>
#include <vector>
#include <algorithm>
#include <ncurses.h>
#include <string>
#include "TaskList.h"

using namespace std;
```

ЯКА КОНСТРУКЦІЯ?

СТВОРИМО БАЗОВИЙ КЛАС "TASK"

```
class Task {
public:
    string title;
    string description;
    string date;
    bool important;
    bool urgent;
```

СТВОРИМО МЕТОД ДРУКУ

```
void printTasksByDate() {
    sort(tasks.begin(), tasks.end(), [](Task const& a, Task const& b) {
        return a.date < b.date;
    });

    for (auto const& task : tasks) {
        cout << task.title << " - " << task.date << " - ";
        if (task.important && task.urgent) {
            cout << "Important and urgent";
        }
        else if (task.important) {
            cout << "Important";
        }
        else if (task.urgent) {
            cout << "Urgent";
        }
        else {
            cout << "Not important and not urgent";
        }
        cout << endl;
    }
}
```

СТВОРИМО ПОХІДНИЙ "TASKLIST"

```
class TaskList {
private:
    vector<Task> tasks;
public:
```

```
void addTask(string title, string description, string date, bool important, bool urgent) {
    Task task;
    task.title = title;
    task.description = description;
    task.date = date;
    task.important = important;
    task.urgent = urgent;
    tasks.push_back(task);
}

void editTask(int index, string title, string description, string date) {
    tasks[index].title = title;
    tasks[index].description = description;
    tasks[index].date = date;
}

void setTaskImportance(int index, bool important) {
    tasks[index].important = important;
}
```

```
void printTasksByImportance() {
    sort(tasks.begin(), tasks.end(), [](Task const& a, Task const& b) {
        if (a.important != b.important) {
            return a.important > b.important;
        }
        else {
            return a.urgent > b.urgent;
        }
    });

    for (auto const& task : tasks) {
        cout << task.title << " - " << task.date << " - ";
        if (task.important && task.urgent) {
            cout << "Important and urgent";
        }
        else if (task.important) {
            cout << "Important";
        }
        else if (task.urgent) {
            cout << "Urgent";
        }
        else {
            cout << "Not important and not urgent";
        }
        cout << endl;
    }
}
```

ДОДАМО МЕТОДИ ДОСТУПУ ДЛЯ ЮЗЕРІВ У "TASKLIST"

```
void setTaskUrgency(int index, bool urgent) {
    tasks[index].urgent = urgent;
}

void removeTask(int index) {
    tasks.erase(tasks.begin() + index);
}

void setTaskImportance(int index, bool important) {
    tasks[index].important = important;
}
```

РЕАЛІЗУЄМО У MAIN

```
int main() {
    initscr();
    noecho();
    cbreak();
    keypad(stdscr, TRUE);
    curs_set(0);

    TaskList taskList;

    int currentTaskIndex = 0;

    while (true) {
        clear();
        mvprintw(2, 0, "Title: ");
        getline(cin, title);
        mvprintw(3, 0, "Description: ");
        getline(cin, description);
        mvprintw(4, 0, "Date (YYYY-MM-DD): ");
        getline(cin, date);
        noecho();

        mvprintw(6, 0, "Is task important (y/n): ");
        if (getch() == 'y') {
            important = true;
        }

        mvprintw(7, 0, "Is task urgent (y/n): ");
        if (getch() == 'y') {
            urgent = true;
        }

        taskList.addTask(title, description, date, important, urgent);
        break;

        case 'i':
        {
            taskList.setTaskImportance(currentTaskIndex, true);
            break;
        }
        case 'u':
        {
            taskList.setTaskUrgent(currentTaskIndex, true);
            break;
        }
        case 'q':
        {
            endwin();
            return 0;
        }
        default:
        {
            break;
        }
    }
}

case 'd':
{
    if (taskList.tasks.size() > 0) {
        taskList.removeTask(currentTaskIndex);
        currentTaskIndex = min(currentTaskIndex, (int)taskList.tasks.size());
    }
    break;
}
```



ЯК ЦЕ ВИГЛЯДАЄ?

```
[_Priori-Time-Management_]
Username: John Doe
Date: 19.03.2023

[ Task List ]
1. Call Grandma (21.03.2023)
2. Buy groceries (20.03.2023)
3. Write report (22.03.2023)

[ [I] Important Task ]
1. Call Grandma (21.03.2023)

[ [U] Urgent Task ]
2. Buy groceries (20.03.2023)

[ [O] Other Task ]
3. Write report (22.03.2023)

[ Select Action ]
1. Add Task
2. Delete Task
3. Change Task Date
4. Change Task Text
5. Mark Task as [...]
6. Show Task List
7. Close Program

Prod. by Anastasiia N.
```

```
[ Task List ]
1. Call Grandma (21.03.2023)
2. Buy groceries (20.03.2023)
3. Write report (22.03.2023)

[ Important Task ]
1. Call Grandma (21.03.2023)

[ Urgent Task ]
2. Buy groceries (20.03.2023)

[ Other Task ]
3. Write report (22.03.2023)

[ Add Task ]
Enter Date:
Enter Task Text:
Enter Priority:
[V] Task added.
[X] Task not added. Please try again.

[ Delete Task ]
Enter Date: 20.03.2023
Enter Priority: U
Is this your task [?]:
2. Buy groceries (20.03.2023)
[YES | NO]

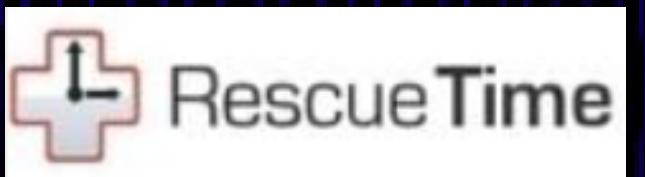
Delete task [?]:
[YES | NO]

[V] Task deleted.
[X] Task not deleted. Please try again.
```

ICHИЮЧИ АНАЛОГИ

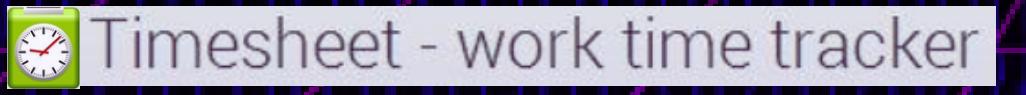


TIME PLANNER



RESCUETIME

MY MINUTES



TIMESHEET

NIRVANA



FOCUS BOOSTER

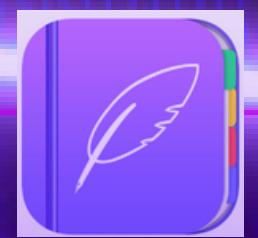


TOGGL

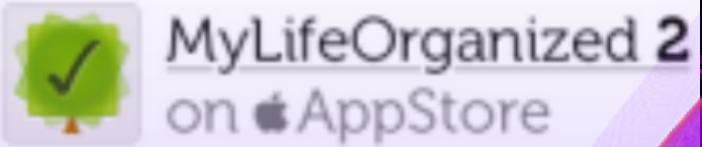


MIND42

MYLIFEORGANIZED (MLO)



DAILY PLANNER



Untitled -TextEdit

File Edit View Help

