# Практически проект: Камък, ножица, хартия

Днес ще разработим конзолна игра "Камък**, ножица, хартия**":

Icon

Description automatically generated

[Камък, ножица, хартия](https://bg.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BC%D1%8A%D0%BA,_%D0%BD%D0%BE%D0%B6%D0%B8%D1%86%D0%B0,_%D1%85%D0%B0%D1%80%D1%82%D0%B8%D1%8F) е **игра за двама**, където вие и вашият опонент (компютъра) едновременно избирате една от следните три опции: **камък** (**rock**), **ножица** (**scissors**) или **хартия** (**paper**). Правилата са следните:

* **Камъкът побеждава ножицата** (ножицата се чупи от камъка)
* **Ножицата побеждава хартията** (хартията се реже от ножицата)
* **Хартията побеждава камъка** (камъкът се покрива от хартията)

**Победител** е потребителят, чийто избор побеждава избора на неговия опонент. Ако и двамата потребители изберат **една и съща опция** (например "хартия"), резултатът от играта е "**draw**" (равен).

A screenshot of a computer

Description automatically generated with medium confidence Graphical user interface, text, application

Description automatically generated

## Създайте GitHub профил и репо

Всеки програмист трябва да има профил в GitHub. Първо, **създайте акаунт в платформата**.

### Регистрация в GitHub

**Регистрирайте се** за безплатен профил в **GitHub** тук: <http://github.com> с имейл и потребителско име:

Graphical user interface, website

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

След като сте готови, е време да **създадете първото си репо (хранилище)**. Едно **репо** съдържа **всички файлове на проекта** и историята на всеки един от файловете.

### Създаване на GitHub репо

Създайте **ново репо** от: <https://github.com/new>. Изберете **смислено име**, например "RockPaperScissorsBy{потребителско име}", добавете **кратко описание** и направете вашето репо **публично**:

Graphical user interface, text, application, Teams

Description automatically generated

|  |  |
| --- | --- |
| Icon  Description automatically generated | Важно е да изберете **ваше оригинално име** за проекта!  Вашият GitHub профил трябва да бъде **уникален** и да не бъде същият като този на вашите съученици.  Може да следвате тези инструкции за разработка на проекта, но можете и да **правите промени** и да **имплементирате проекта по различен начин** от вашите съученици. |

Също **добавете** README.md файл и .gitignore **за Visual Studio**, както е показано:

Graphical user interface, application, Teams

Description automatically generated

В Git проекти файлът .gitignore уточнява кои файлове от репото **не са** част от source кода и трябва да бъдат **игнорирани** (не трябва да бъдат качвани в репото). Обикновено в GitHub репото искаме да качваме **само source кода**, без компилирани бинарни и временни файлове.

**Променете лиценза** на "MIT" (който е най-разпространеният open source лиценз) или друг лиценз по избор и натиснете **бутона**  [Create], за да **създадете репото**:

Graphical user interface, text, application, email

Description automatically generated

Сега вашето **репо е създадено** и изглежда по подобен начин:

Graphical user interface, text, application, email, website

Description automatically generated

Сега е време да **напишете кода** на играта.

## Код за играта

### Създаване на проект във Visual Studio

Първо, **стартирайте Visual Studio** и **създайте нов C# console application**:

Graphical user interface, application

Description automatically generated

След това **избирате име** и **място за съхранение на проекта**. Трябва да **отбележите** кутийката с "**Place solution and project in the same directory**", за да не се налага да правите допълнителна папка за файловете. След това изберете [Next]:

Graphical user interface, text, application, email

Description automatically generated

След това изберете [.NET 6 (Long-term support)] и кликнете **[Create]**:

Graphical user interface, text, application

Description automatically generated

**Проектът трябва да е създаден** и да изглежда така:

Graphical user interface, text, application, Word

Description automatically generated

Преди да продължите, е важно да **смените името** на главния клас – Program.cs – с нещо **по-смислено**. Можем да го направите чрез **десен клик** върху файла → [**Rename**]:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

### Имплементиране на логиката на играта

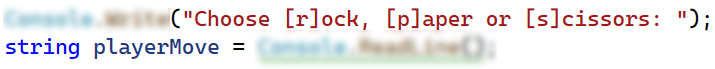
#### Прочитане на избора на потребителя

Създайте **три константи** за "**Rock**" (камък), "**Paper**"(хартия) и "**Scissors**" (ножица), които ще използваме по-късно. **Константите** са стойности, които **не се променят** в програмата. Трябва да изглеждат така:

A screenshot of a computer

Description automatically generated with medium confidence

След това напишете в конзолата опциите **("rock", "paper", "scissors")**, които потребителят може да избере и прочетете **входните данни**. Вече сте учили как да го направите:



Сега можете да изпълните приложението в конзолата и да проверите дали кодът работи правилно:

Graphical user interface

Description automatically generated with medium confidence Text

Description automatically generated

Можете да видите текста, **изписан** на конзолата, и можете да **въведете избраната опция**.

#### Свързване на хода на потребителя с някоя от възможните опции

Сега е време да свържете потребителския вход с някоя от **възможните опции**. За да направите това, създайте **if-else** конструкция с **възможните ходове** и променете стойността на променливата с някоя от създадените константи.

Първо, ако потребителят е въвел **"r"** или **"rock"**, той е избрал **"Rock"**. Можете да напишете кода така:

Text

Description automatically generated

Ако е въвел **"p"** или **"s"**, значи е избрал **"Paper"** или **"Scissors"**. Напишете сами else-if конструкцията:

Graphical user interface, text

Description automatically generated

Сега трябва да се погрижите за варианта, в който потребителят въвежда **невалидна стойност**. За да направите това, можете да използвате **else**, да **отпечатате** съобщение на конзолата и да **прекратите изпълнението на програмата**:

Logo, company name

Description automatically generated

Сега може да **изпълните приложението** в конзолата и да проверите дали този код **работи правилно**, когато въведете **невалидна стойност**.

Text

Description automatically generated

#### Избор на ход на компютъра

Направете **променлива от тип Random**, която ще помогне да **изберете случайно число**, използвайки метода Next(). Ще използвате това **число**, за да направите избор за компютъра между "rock", "paper" и "scissors":

Text

Description automatically generated with low confidence

Можете да научите повече за **Random** тук: <https://docs.microsoft.com/en-us/dotnet/api/system.random.next?view=net-6.0> .

Ще се нуждаете от **променлива от тип string**, за да съхраним **хода на компютъра**:

Text

Description automatically generated with low confidence

За да свържем „избора“ на компютъра с някоя от **възможните опции**, можем да използваме switch-case или else-if:

Graphical user interface, text

Description automatically generated with medium confidence

Помислете как можете да довършите тази конструкция.

#### Проверка и отпечатване на резултата

Отпечатайте на конзолата какъв е **случайният (random)** избор на компютъра, например: "**The computer chose {ход на компютъра}.**".

Сега трябва да **сравните** избора на **потребителя** с този на **компютъра**, използвайки **условна конструкция**.

Text

Description automatically generated

Можете да използвате тази таблица с **възможни комбинации**:

Table

Description automatically generated

Обмислете **всички варианти**, в които потребителят **губи** или резултатът е **равен** и напишете условни конструкции. Това е всичко необходимо, за да създадете **работеща игра**.

Graphical user interface, text, application, email

Description automatically generated

След като изпълните програмата, играта трябва да изглежда така:

Graphical user interface, text

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text

Description automatically generated

## Качване на проекта в GitHub

Сега е време да качите проекта в **GitHub**, за да може и останалите да го видят и ако искат, да го тестват, да могат да го клонират и да го стартират локално. Има **две опции**, изберете една и следвайте стъпките:

### Използвайки TortoiseGit (Опция 1)

Ако нямате TortoiseGit на компютъра си, първо го свалете и го инсталирайте от <https://tortoisegit.org>.

Използвайте **Git clone** за клониране с TortoiseGit. Отидете в директорията, в която искате да го съхраните, натиснете **десен бутон** в папката и изберете [Git Clone**]**.

Graphical user interface, application

Description automatically generated

Сега отидете във вашия **GitHub** профил, отворете новосъздаденото **репо** и копирайте неговия **repo URL**. Първо кликнете на [Repositories]:

A screenshot of a computer

Description automatically generated with medium confidence

След това изберете репото на проекта

Graphical user interface, application

Description automatically generated

Сега кликнете на [Code] и копирайте **URL-а** на **репото**.

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

Върнете се в **TortoiseGit**, поставете **URL**-а и кликнете [OK]:

Graphical user interface, application

Description automatically generated

The results should be something like this:

Graphical user interface, text, application

Description automatically generated

Файловете от вашето GitHub репо ще бъдат свалени в **подпапка** с името на проекта в GitHub, в случая "**RockPaperScissorsByPeter**".

Graphical user interface, text, application

Description automatically generated

Когато отворите папката, която съдържа клонираното репо, трябва да изглежда така:

Graphical user interface, text, application

Description automatically generated

Следващото нещо, което трябва да направите, е да **добавите проекта** към **клонираното репо**. Можете да използвате "**Cut & Paste**". Трябва да изглежда така:

Graphical user interface, application

Description automatically generated

Сега можете да **качите** промените на проекта в GitHub.

Можете да използвате [GitCommit…]от TortoiseGit. Отидете в папката на проекта, натиснете **десен бутон** и кликнете [Git Commit -> "main"…].

Graphical user interface, text, application

Description automatically generated

Добавете **подходящо** съобщение и кликнете [Add], за да не пропуснете някои файлове, и след това изберете [Commit].

Graphical user interface, application

Description automatically generated

Резултатът трябва да изглежда така:

Graphical user interface, text, application

Description automatically generated

След това кликнете [Push] и после [OK].

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Резултатът трябва да е следният:

Graphical user interface, text, application

Description automatically generated

Това е всичко, от което се нуждаете, за да **качите вашия проект** във вашето **GitHub репо**, използвайки TortoiseGit.

### Използвайки Git Bash (Опция 2)

Като **алтернатива на предишния вариант**, ако нямате **TortoiseGit**, можете да използвате"**Git Bash**", за да качите проекта си във вашето GitHub репо.

Първо, ако нямате Git на своя **компютър**, го **инсталирайте** от <https://git-scm.com/downloads>.

Отидете в **директорията** с проекта, натиснете **десен бутон** в папката и изберете [**Git Bash Here**], за да отворите конзолата на Git. Ако менюто [**Git Bash Here**] **липсва**, първо трябва да инсталирате **Git**.

Graphical user interface, application

Description automatically generated

Напишете командата **"**gitclone**"**, последвана от линка на **репото**:

|  |
| --- |
| gitclone |

Поставете **URL**-а към вашето репо след командата.

Graphical user interface, text

Description automatically generated

Резултатът трябва да е следният:

Text

Description automatically generated

Файловете от вашето репо ще бъдат свалени в **подпапка** с името на самия проект в GitHub, в този случай "**RockPaperScissorsByPeter**".

Graphical user interface, text, application

Description automatically generated

Когато отворите папката, тя трябва да изглежда така:

Graphical user interface, text, application

Description automatically generated

Следващото нещо, което трябва да направите, е да **добавите файловете от проекта** в **клонираната папка**. Трябва да изглежда така:

Graphical user interface, application

Description automatically generated

Сега сте готови да качите промените. Отидете в папката на репото, натиснете **десен бутон**, изберете"GitBashHere" и изпълнете следните **команди**.

|  |
| --- |
| git status |

**Git status** показва състоянието на директорията.

Text

Description automatically generated

Сега напишете:

|  |
| --- |
| git add . |

Горната команда **добавя** всички нови/модифицирани файлове към вашето локално **Git repo**.

Graphical user interface

Description automatically generated

Сега напишете:

|  |
| --- |
| git commit -m "Uploaded my first project" |

Тази команда **commit**-ва вашите промени във вашето локално **Git repo**. Също трябва да **добавите** подходящо **съобщение**.

Text

Description automatically generated

Остава да изпълните **още две** **команди**. Напишете:

|  |
| --- |
| git pull |

Тази команда **обновява** вашето локално **репо** от GitHub. Сваля последната версия на проекта от GitHub и я слива с вашето локално копие.

Graphical user interface, application

Description automatically generated

Последното нещо, което трябва да направите, е да **push**-нете промените чрез следната команда:

|  |
| --- |
| git push |

Тази команда **качва вашите локални промени в GitHub**.

Text

Description automatically generated

Това е всичко, от което се нуждаете, за да **обновите** вашето **репо**, използвайки **Git Bash**.

Можете да прочетете **повече за** **Git Bash** тук: <https://git-scm.com/about>.

## \* Модифициране на кода, добавяне на нови функционалности

Сега може да си **поиграете с кода** и **да го модифицирате**.

|  |  |
| --- | --- |
| Icon  Description automatically generated | Това е ваш собствен проект. **Бъдете уникални**.   * Имплементирайте **свои собствени функционалности**. * Имплементирайте кода **сами**, добавете ваш стил, форматиране, коментари и т.н. |

Ето няколко **идеи**, които можете да имплементирате.

### Добавяне на цветове

Можете да промените **цвета** на **текста** и на **фона** в конзолата:   
<https://www.c-sharpcorner.com/article/change-console-foreground-and-background-color-in-c-sharp>.

Graphical user interface, text, application

Description automatically generated Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated Text

Description automatically generated

### Рестартиране на играта

Можете автоматично да **рестартирате играта**, след като е завършила (или да попитате участника дали иска да играе отново).

Text

Description automatically generated Text

Description automatically generated

### Система за оценяване

Можете да добавите **система за оценяване**, която изписва броя на победите на потребителя и на компютъра след всяка игра.

### Допълнителни идеи

Можете да промените логиката, така че да **увеличите шансовете за победа на потребителя**.

### Commit-ване в GitHub

Сега можете да **commit**-нете и **push**-нете промените си в **GitHub**.

|  |  |
| --- | --- |
| Icon  Description automatically generated | Много е важно да **commit-вате често** вашия код в GitHub. По този начин създавате **богата commit история** за вашия проект:  A picture containing chart  Description automatically generated |

## Създаване на README.md файл

Силно препоръчително е да напишете **документация за вашия проект в GitHub**, за да опишете за какво е предназначен. Започнете като редактирате файла README.md от вашето репо:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Добавете име на проекта. Използвайте "#" пред текста, за да означите, че е **заглавие**:

Graphical user interface, application

Description automatically generated

Можете да **видите** прогреса си, като натиснете бутона [Preview]:

Graphical user interface, application

Description automatically generated 🡪Graphical user interface, application, Word

Description automatically generated

### Секции на документацията

Добавете **информация** за вашия проект във вашия README.md файл. Обикновено документацията съдържа следните **секции**:

* **Заглавие на проекта** (трябва да отговаря на въпроса "Какво има в проекта?")
* **Цели на проекта**  (какъв проблем решаваме, напр. имплементация на игра)
* **Решение**  (трябва да описва как решаваме проблема 🡪 **алгоритми**, **технологии**, **библиотеки** и т.н.)
* **Линк към source код**
* **Скрийншоти** (скрийншоти към различните функционалности на проекта)
* **Демо наживо** (линк към директно демо на проекта)

### Използвайте markdown

Забележете, че README.md файла се пише на **Markdown език**. Markdown комбинира текст и специално форматирани тагове.

Можете да научите повече за **Markdown** тук: <https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax>.

### Цели на проекта

Започнете документацията, като опишете **целите на проекта**. Какъв проблем решава проектът?

### Примерна документация

Това е **пример** как може да документирате проекта си. Съветваме ви да не го преписвате, а **сами да структурирате документацията си**!

Graphical user interface, text, application, email

Description automatically generated

Можете да добавите **подходящи изображения**, за да подобрите документацията. Можете да добавите изображение по следния начин:



Добавете информация за **входа** и **изхода** на проекта:

Graphical user interface, text, application, chat or text message, email

Description automatically generated

### Вашето решение

Опишете как **решавате** проблема: **алгоритми**, **технологии**, **библиотеки** и **т.н.**

Например за нашата игра можете да анализирате всички възможни ситуации в таблица:

Table

Description automatically generated

### Линк към сорс код

Добавете **линк** към вашия **сорс код**, както следва:



### Скрийншоти

Добавете **скрийншоти** на вашия проект:

1. **Направете скрийншот** с предназначен за това инструмент (например [Snipping Tool](https://support.microsoft.com/en-us/windows/open-snipping-tool-and-take-a-screenshot-a35ac9ff-4a58-24c9-3253-f12bac9f9d44) за Windows).

Graphical user interface, text, application, email

Description automatically generated

1. **Поставете** скрийншота във файла за документация, използвайки [Ctrl+V]:

Graphical user interface, text, application, email

Description automatically generated

Примерни скрийншоти за играта "Rock Paper Scissors":

Text

Description automatically generated

## Качване на приложението си в Replit

Replit e онлайн среда за разработка (online IDE), която позволява да **пишете** софтуерни проекти, да ги **споделяте** чрез линк и да ги **изпълнявате** директно в браузъра.

Създайте си профил в Replit, за да можете да показвате **проектите си** на вашите приятели и да сложите линка в секцията "**live demo links**" във вашата GitHub документация. Създайте **безплатен** **Replit** акаунт тук: [https://replit.com](https://replit.com/).

Graphical user interface, text, website

Description automatically generated

A screenshot of a computer

Description automatically generated

Създайте **нов проект** в Replit от **менюто** в **горния** **ляв ъгъл**.

Graphical user interface, text, application, chat or text message, website

Description automatically generated

Кликнете [Create], след това изберете **езика**, на който е **написан** проектът, изберете **име** и **създайте** проекта.

Graphical user interface, text, application

Description automatically generated

Изберете "Mono C#".

Graphical user interface, text, application, chat or text message

Description automatically generated

Добавете **смислено име** за вашия Replit проект, например "**RockPaperScissors-Game-by-Peter**".

**Поставете вашия код** във файла "Main.cs":

Graphical user interface, text, application

Description automatically generated

**Кликнете** [Run] и ще изпълните приложението си директно в браузъра:

Text

Description automatically generated

## Добавете Replit линк към вашия README.md файл

Сега добавете **линк към демо** от вашия проект към вашата GitHub документация:

Text

Description automatically generated

Можете да направите **скрийншот** от Replit.com и **да го поставите** в GitHub документацията директно чрез **[Ctrl+V]**.

След промените документацията трябва да изглежда така:

Graphical user interface, text, application, chat or text message

Description automatically generated

Сега, когато кликнете върху бутона [**Run**], ще бъдете пренасочени към демото в Replit.

 🡪

Graphical user interface, application

Description automatically generated

Поздравления! Успешно създадохте вашата **първа конзолна игра** и добавихте вашия **първи проект** към вашето GitHub портфолио.