Рекурсия

Трифон Трифонов

Увод в програмирането,
спец. Компютърни науки, 1 поток,
спец. Софтуерно инженерство,
2016/17 г.

21 декември 2016 г. – 4 януари 2017 г.

Какво е рекурсия?

Какво е рекурсия?

N. Wirth, Algorithms and Data Structures, Fig 3.1

Какво е рекурсия?

Какво е рекурсия?

- Повторение чрез позоваване на себе си

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”
- директориите съдържат файлове и директории

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”
- директориите съдържат файлове и директории
- PHP = PHP Hypertext preprocessor

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”
- директориите съдържат файлове и директории
- PHP = PHP Hypertext preprocessor
- за да строшите камък:

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”
- директориите съдържат файлове и директории
- PHP = PHP Hypertext preprocessor
- за да строшите камък:
  - ударете с чука, за да натрошите камъка на части

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- “приятелите на моите приятели са и мои приятели”
- директориите съдържат файлове и директории
- PHP = PHP Hypertext preprocessor
- за да строшите камък:
  - ударете с чука, за да натрошите камъка на части
  - строшете получените по-малки камъни

Какво е рекурсия?

- Повторение чрез позоваване на себе си
- "приятелите на моите приятели са и мои приятели"
- директориите съдържат файлове и директории
- PHP = PHP Hypertext preprocessor
- за да строшите камък:
  - ударете с чука, за да натрошите камъка на части
  - строшете получените по-малки камъни
- за да разберете какво е рекурсия, трябва да разберете какво е рекурсия

Рекурсията в математиката

$$n! = \begin{cases} 1, & n = 0, \\ n(n-1)!, & n > 0. \end{cases}$$

Рекурсията в математиката

$$n! = \begin{cases} 1, & n = 0, \\ n(n-1)!, & n > 0. \end{cases}$$

$$x^n = \begin{cases} 1, & n = 0, \\ x.x^{n-1}, & n > 0, \\ \frac{1}{x^{-n}}, & n < 0. \end{cases}$$

Рекурсията в математиката

$$n! = \begin{cases} 1, & n = 0, \\ n(n-1)!, & n > 0. \end{cases}$$

$$x^n = \begin{cases} 1, & n = 0, \\ x.x^{n-1}, & n > 0, \\ \frac{1}{x^{-n}}, & n < 0. \end{cases}$$

$$gcd(a,b) = \begin{cases} a, & a = b, \\ gcd(a-b,b), & a > b, \\ gcd(a,b-a), & a < b. \end{cases}$$

Рекурсията в математиката

$$n! = \begin{cases} 1, & n = 0, \\ n(n-1)!, & n > 0. \end{cases}$$

$$x^n = \begin{cases} 1, & n = 0, \\ x.x^{n-1}, & n > 0, \\ \frac{1}{x^{-n}}, & n < 0. \end{cases}$$

$$gcd(a,b) = \begin{cases} a, & a = b, \\ gcd(a-b,b), & a > b, \\ gcd(a,b-a), & a < b. \end{cases}$$

$$f(x) = \begin{cases} 0, & x = 0, \\ f(x+1) - 1, & x > 0. \end{cases}$$

Как се решават задачи с рекурсия?

- Декомпозиция — свеждане на дадена задача към множество от по-прости задачи

Как се решават задачи с рекурсия?

- Декомпозиция — свеждане на дадена задача към множество от по-прости задачи
- Рекурсията е вид декомпозиция, при който свеждаме задача към множество от по-прости задачи подобни на първоначалната

Как се решават задачи с рекурсия?

- Декомпозиция — свеждане на дадена задача към множество от по-прости задачи
- Рекурсията е вид декомпозиция, при който свеждаме задача към множество от по-прости задачи подобни на първоначалната
- Как работи:

Как се решават задачи с рекурсия?

- Декомпозиция — свеждане на дадена задача към множество от по-прости задачи
- Рекурсията е вид декомпозиция, при който свеждаме задача към множество от по-прости задачи подобни на първоначалната
- Как работи:
  - Показваме решението на най-простите задачи (база, дъно)

Как се решават задачи с рекурсия?

- Декомпозиция — свеждане на дадена задача към множество от по-прости задачи
- Рекурсията е вид декомпозиция, при който свеждаме задача към множество от по-прости задачи подобни на първоначалната
- Как работи:
  - Показваме решението на най-простите задачи (база, дъно)
  - Показваме как по-сложна задача се свежда към една или няколко по-прости (стъпка)

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n + 1)$.

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n + 1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n + 1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓
- нека допуснем, че сме доказали свойството за дадено n

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n + 1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓
- нека допуснем, че сме доказали свойството за дадено n
- ще го докажем за $n + 1$:

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n + 1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓
- нека допуснем, че сме доказали свойството за дадено n
- ще го докажем за $n + 1$:
- $(2+4+\ldots+2n)+2(n+1) = n(n+1)+2(n+1) = (n+1)(n+2)$ ✓

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n+1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓
- нека допуснем, че сме доказали свойството за дадено n
- ще го докажем за $n + 1$:
- $(2+4+\ldots+2n)+2(n+1) = n(n+1)+2(n+1) = (n+1)(n+2)$ ✓
- **Следователно:** доказахме свойството за произволно n. □

Математическа индукция

Дефиниция

Математическата индукция е метод за доказателство, използващ като предпоставка свойството, което се доказва.

Пример: Да се докаже, че $2 + 4 + \ldots + 2n = n(n+1)$.

Доказателство:

- за $n = 0$: трябва да проверим, че $0 = 0.1$ ✓
- нека допуснем, че сме доказали свойството за дадено n
- ще го докажем за $n + 1$:
- $(2+4+\ldots+2n)+2(n+1) = n(n+1)+2(n+1) = (n+1)(n+2)$ ✓
- **Следователно:** доказахме свойството за произволно n. □

Математическата индукция е рекурсивен метод за доказателство.

Рекурсията в програмирането

Дефиниция

Рекурсивна функция наричаме функция, която извиква себе си пряко или косвено.

Рекурсията в програмирането

Дефиниция

Рекурсивна функция наричаме функция, която извиква себе си пряко или косвено.

Рекурсивни функции се поддържат от почти всички съвременни езици за програмиране.

Рекурсията в програмирането

Дефиниция

Рекурсивна функция наричаме функция, която извиква себе си пряко или косвено.

Рекурсивни функции се поддържат от почти всички съвременни езици за програмиране.

Теорема

Всяка програма с цикли може да се напише с рекурсия и обратно.

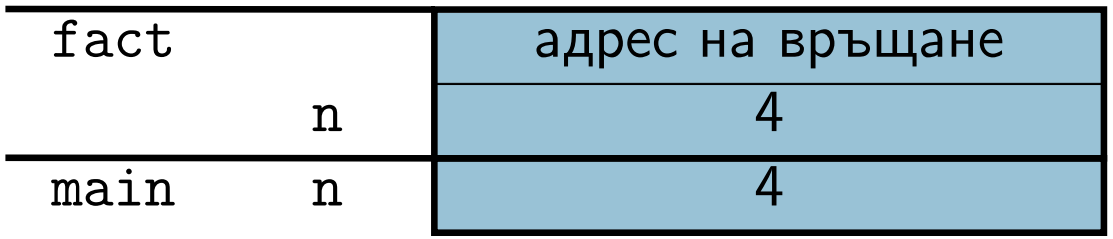
Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$

Стекови рамки на рекурсивни функции

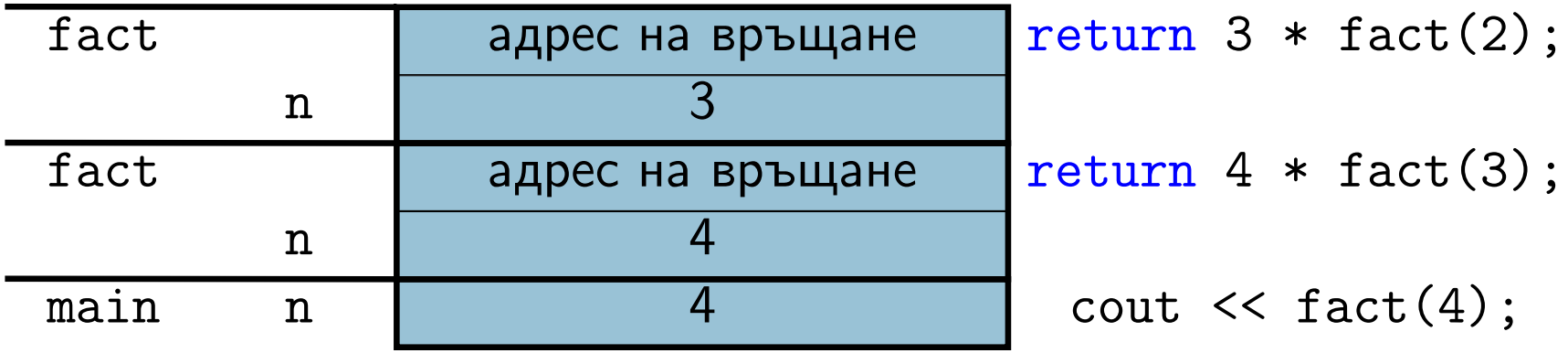
Стекови рамки на рекурсивни функции



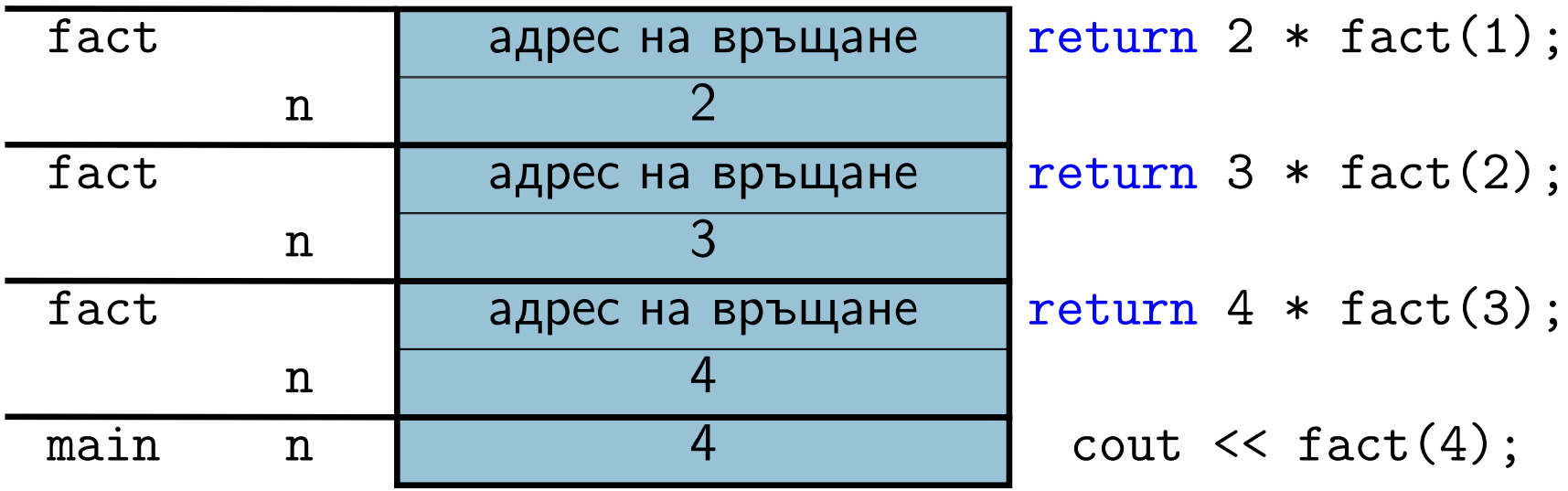

```
return 4 * fact(3);

cout << fact(4);
```

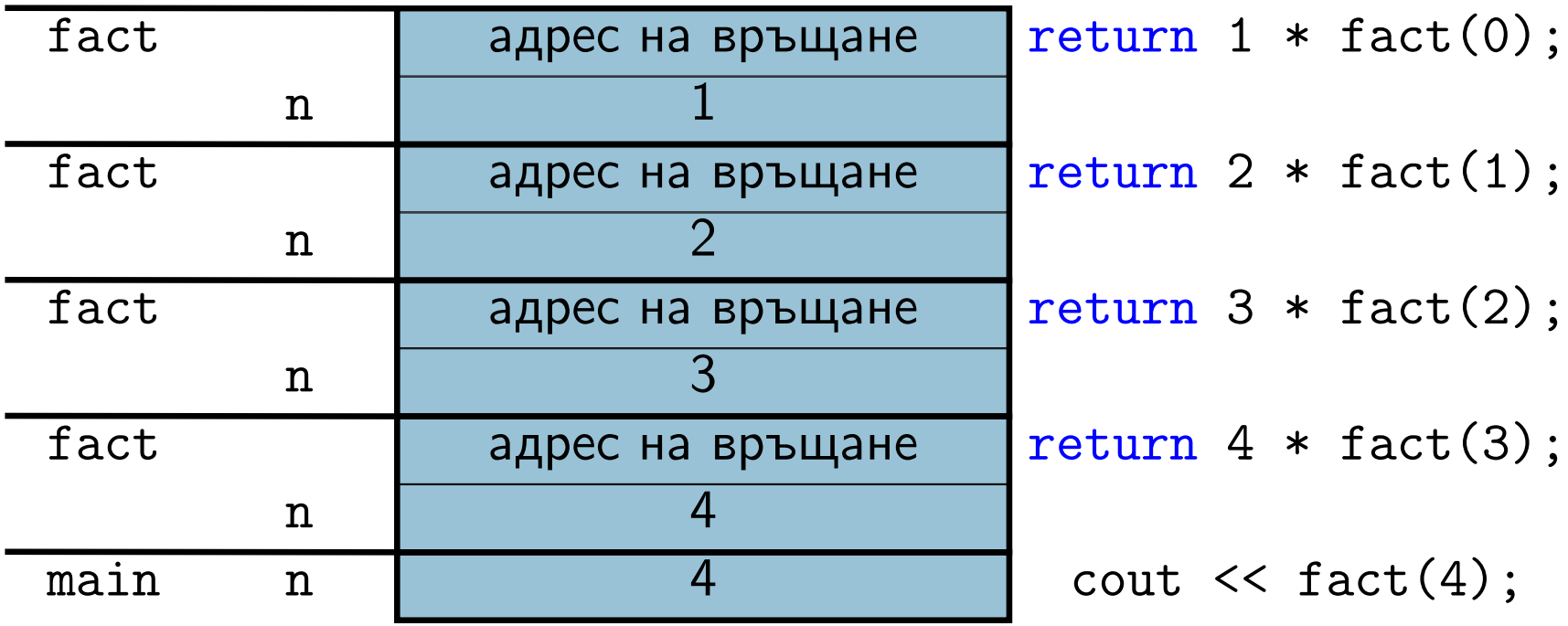
Стекови рамки на рекурсивни функции



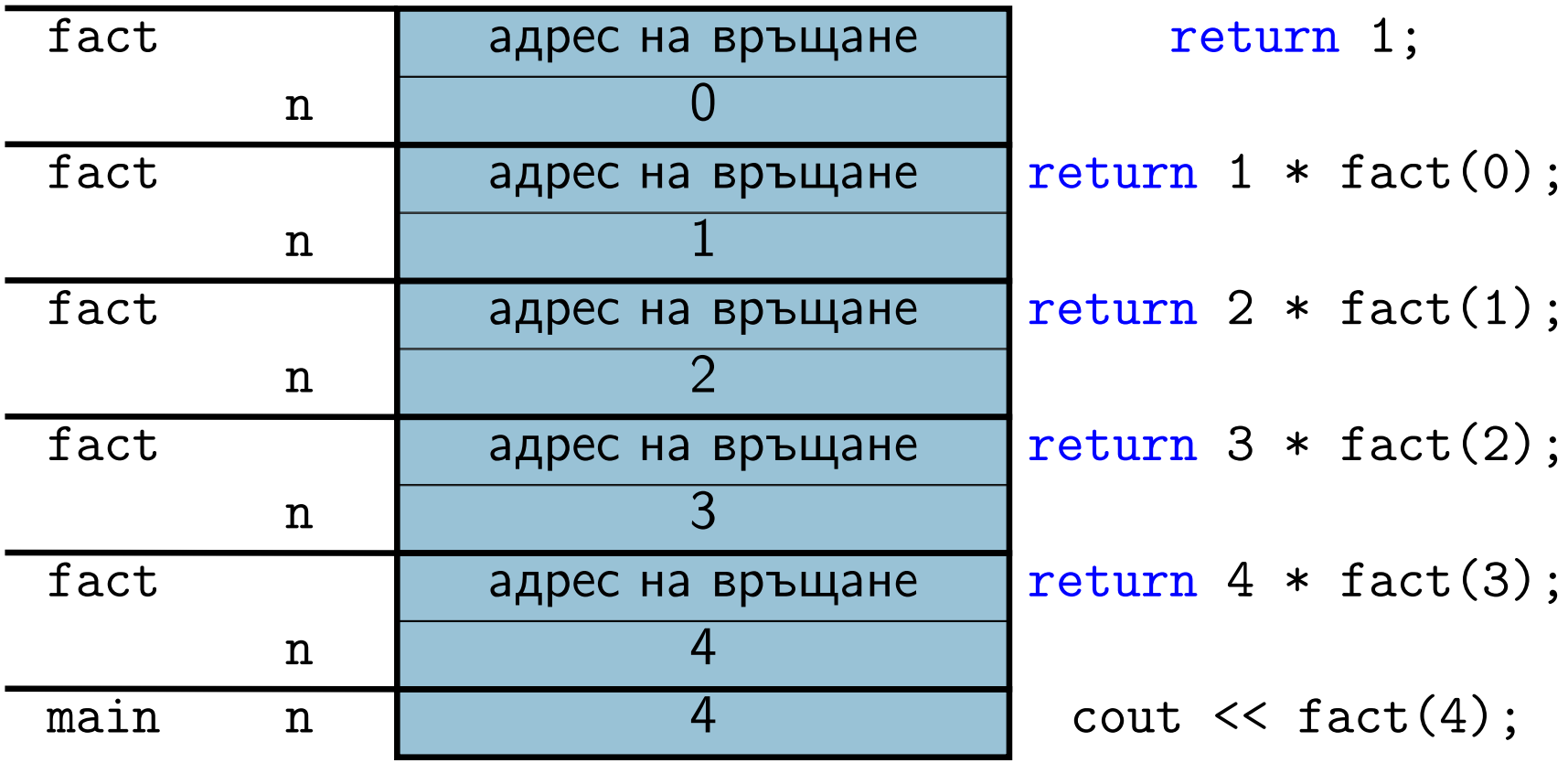
Стекови рамки на рекурсивни функции



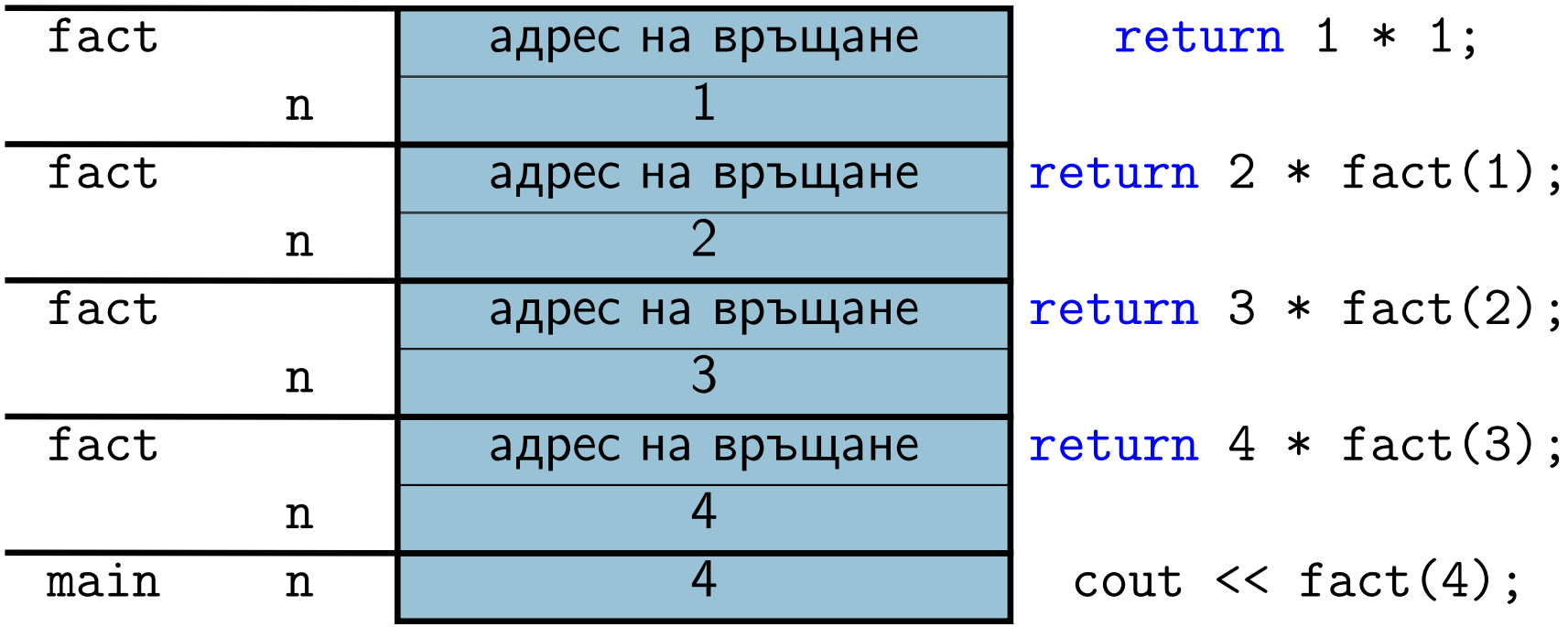
Стекови рамки на рекурсивни функции



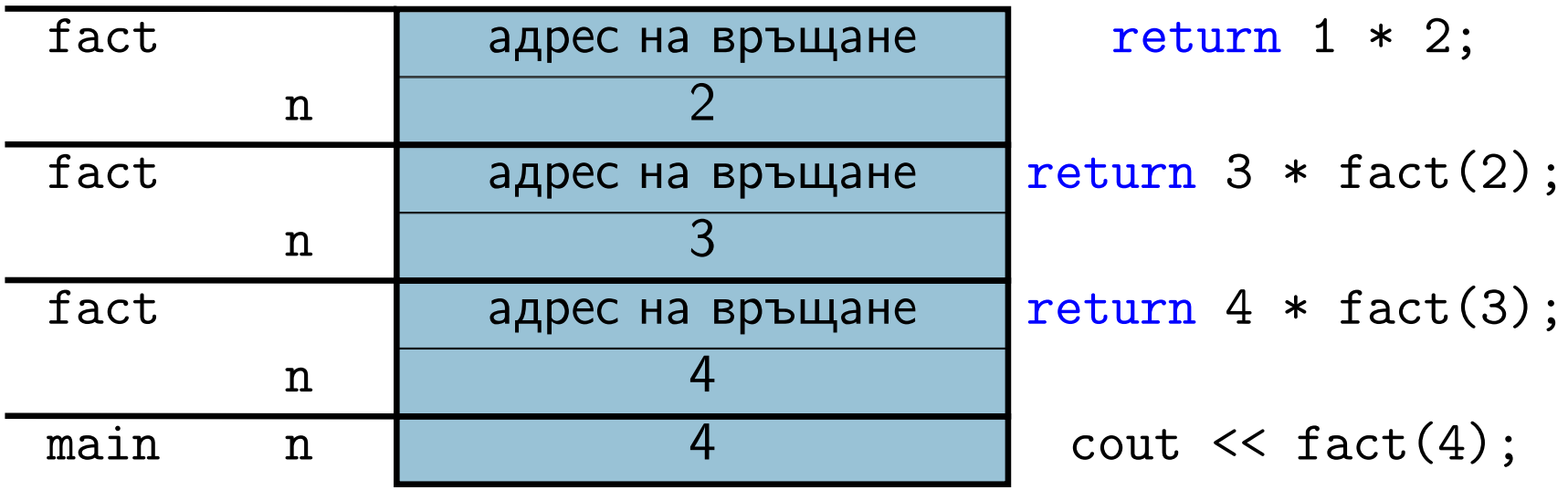
Стекови рамки на рекурсивни функции



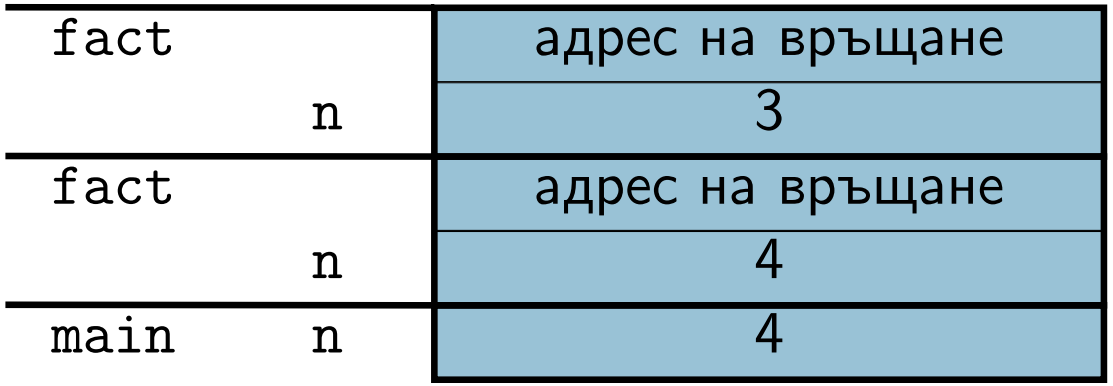
Стекови рамки на рекурсивни функции



Стекови рамки на рекурсивни функции



Стекови рамки на рекурсивни функции

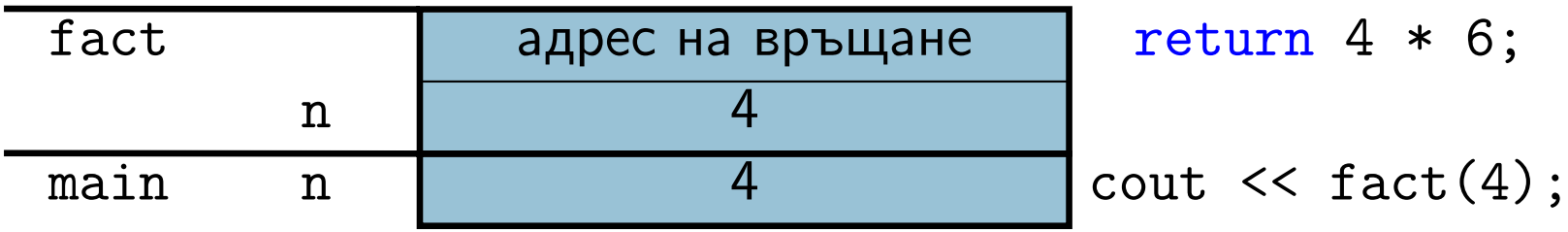



```
return 3 * 2;

return 4 * fact(3);

cout << fact(4);
```

Стекови рамки на рекурсивни функции



Стекови рамки на рекурсивни функции

Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$
2. НОД

Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$
2. НОД
3. x^n

Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$
2. НОД
3. x^n
4. числата на Фибоначи

Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$
2. НОД
3. x^n
4. числата на Фибоначи
5. числата на Фибоначи, но по-бързо.

Примери за рекурсивни функции

Да се напише функция, която пресмята рекурсивно:

1. $n!$
2. НОД
3. x^n
4. числата на Фибоначи
5. числата на Фибоначи, но по-бързо.
6. <израз> със скоби, където
   - <израз> ::= <цифра> | (<израз><операция><израз>)
   - <цифра> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
   - <операция> ::= + | - | * | /

Рекурсивни функции за масиви

Да се напише функция, която чрез рекурсия

1. намира сума на елементите на масив

Рекурсивни функции за масиви

Да се напише функция, която чрез рекурсия

1. намира сума на елементите на масив
2. проверява дали елемент съществува в масив

Рекурсивни функции за масиви

Да се напише функция, която чрез рекурсия

1. намира сума на елементите на масив
2. проверява дали елемент съществува в масив
3. проверява дали елементите на масив са подредени в растящ ред

Рекурсивни функции за масиви

Да се напише функция, която чрез рекурсия

1. намира сума на елементите на масив
2. проверява дали елемент съществува в масив
3. проверява дали елементите на масив са подредени в растящ ред
4. проверява дали елементите на масив са различни

Рекурсивни функции за масиви

Да се напише функция, която чрез рекурсия

1. намира сума на елементите на масив
2. проверява дали елемент съществува в масив
3. проверява дали елементите на масив са подредени в растящ ред
4. проверява дали елементите на масив са различни
5. сортира масив с алгоритъма за “бързо сортиране”

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:
   - елементи по-малки от оста

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:
   - елементи по-малки от оста
   - елементи по-големи или равни на оста

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:
   - елементи по-малки от оста
   - елементи по-големи или равни на оста
3. поставяме оста между двете части на масива

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:
   - елементи по-малки от оста
   - елементи по-големи или равни на оста
3. поставяме оста между двете части на масива
4. рекурсивно сортираме поотделно двете части на масива

Алгоритъм за бързо сортиране

1. Избираме елемент от масива (“ос”)
2. Разделяме масива на две части:
   - елементи по-малки от оста
   - елементи по-големи или равни на оста
3. поставяме оста между двете части на масива
4. рекурсивно сортираме поотделно двете части на масива

Този подход за решение се нарича “разделяй и владей”.