C2110 *Operační systém UNIX a základy programování*

3. lekce / modul 2

PS/2020 Distanční forma výuky: Rev1

Petr Kulhánek

kulhanek@chemi.muni.cz

Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta
Masarykova univerzita, Kamenice 5, CZ-62500 Brno

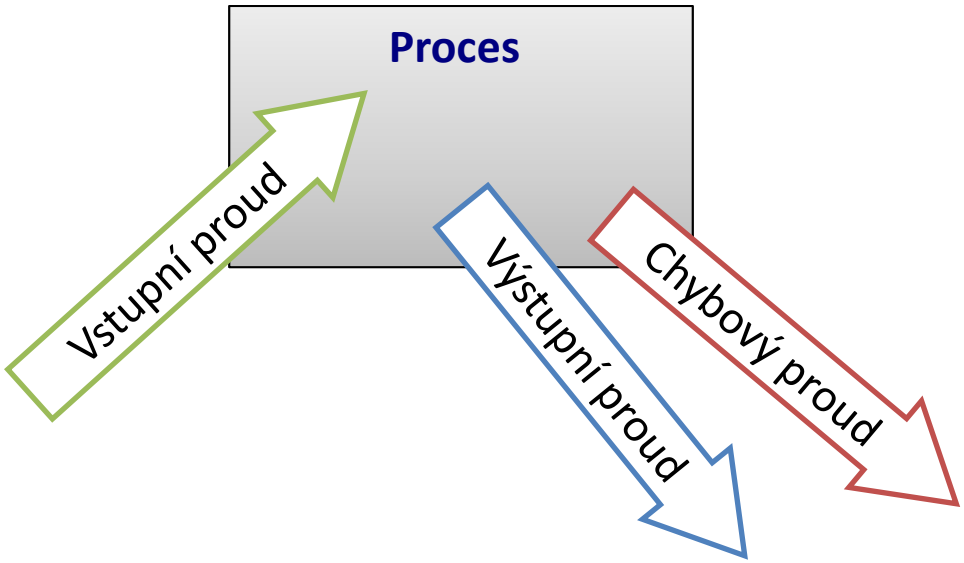
Proudy, přesměrování, roury

Komunikace procesu s okolím

Proces může komunikovat s okolím celou řadou způsobů:

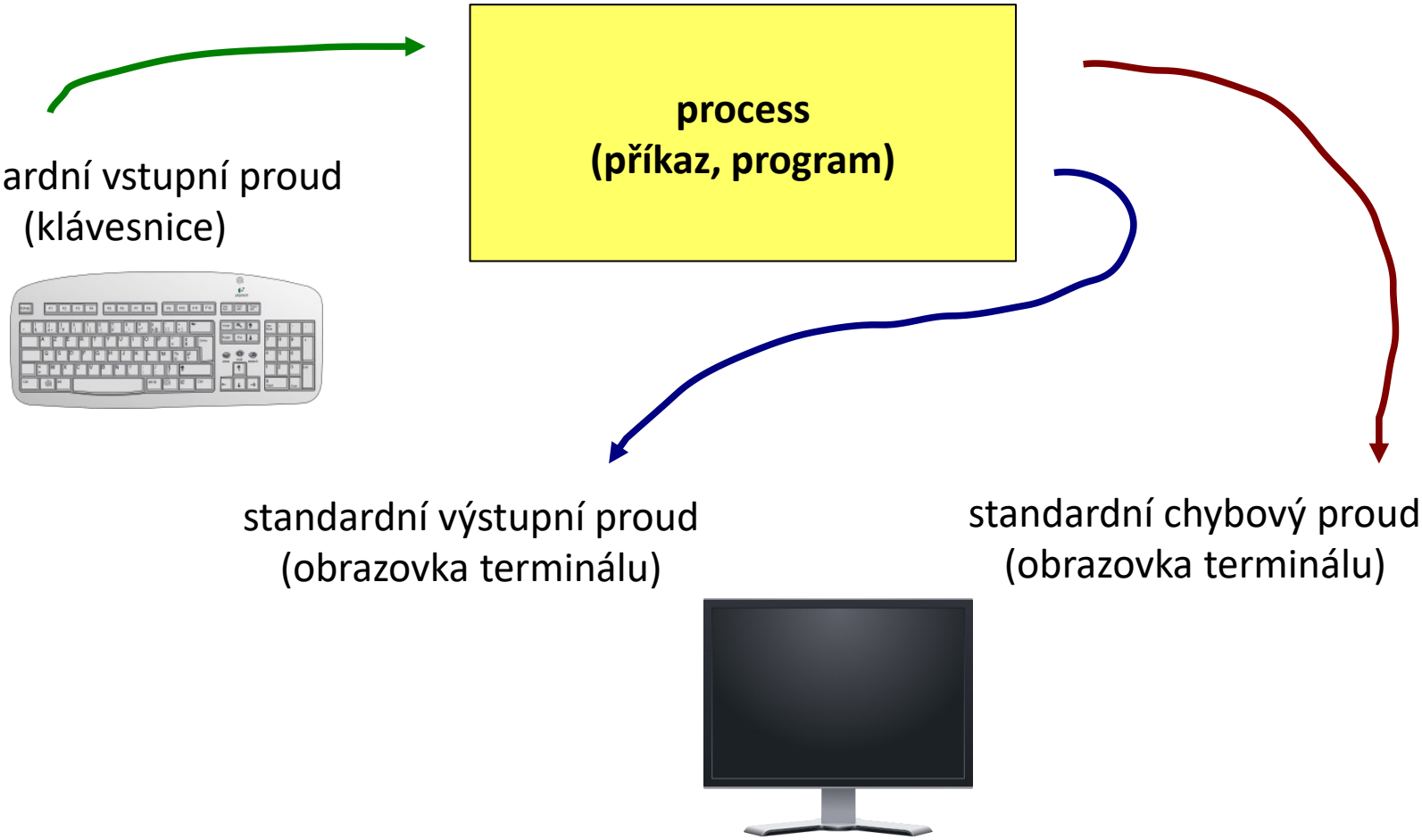
- GUI (Graphical User Interface = použitím příslušného API)
- signály, sdílená paměť, MPI (Message Passing Interface), atd.
- standardní proudy

Jednou z možností je načítaní vstupních dat ze **standardního vstupního proudu**, výpis výstupních dat do **standardního výstupního** či **chybového proudu**.



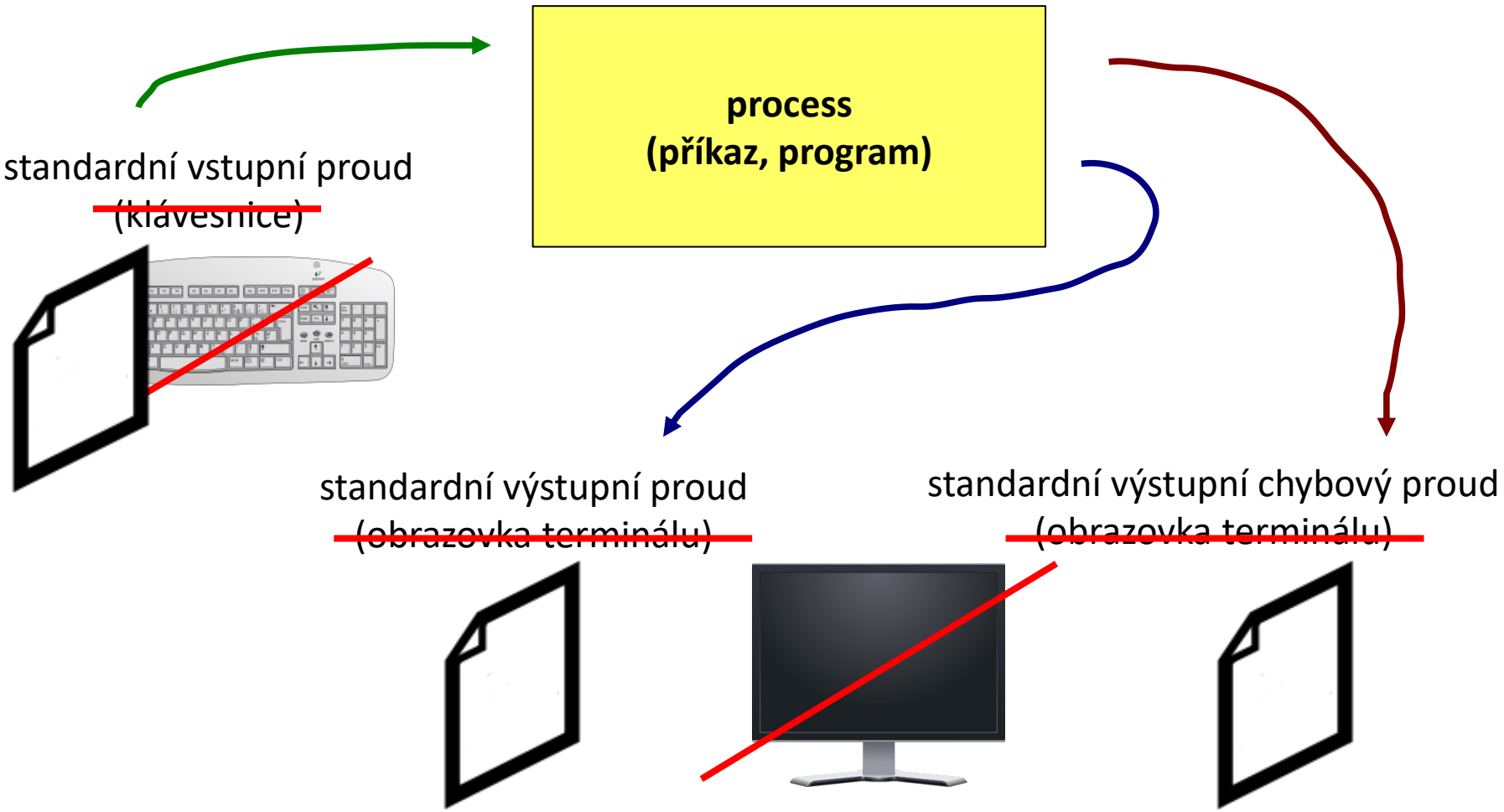
Standardní proudy

Vstupně-výstupní proudy slouží procesu ke **komunikaci** se svým okolím. Každý proces otevírá **tři standardní proudy**:



Přesměrování

Vstupně-výstupní proudy lze přesměrovat tak, aby používaly **soubory** místo klávesnice či obrazovky.



Přesměrování a ukončení vstupu

Přesměrování standardního vstupu programu my_command ze souboru **input.txt**.

```
$ my_command < input.txt
```

Přesměrování standardního vstupu programu my_command ze souboru skriptu.

```
.......
./my_command << EOF
prvni radka textu
druha radka textu
treti radka textu
EOF
......
```

značka určující konec vstupu (volí uživatel)

text, který tvoří načítaný vstup

konec vstupu, *značku **nesmí** obklopovat mezery*

Tento způsob přesměrování je obzvláště výhodné používat ve skriptech, nicméně funguje i v příkazové řádce. Výhodou je expanze proměnných v načítaném textu.

Terminál (užitečné klávesové zkratky):

Ctrl+D zavře vstupní proud spuštěného procesu

Přesměrování výstupu

Přesměrování standardního výstupu programu my_command do souboru **output.txt**. (Soubor output.txt je vytvořen. Pokud již existuje, je jeho původní obsah **smazán**.)

```
$ my_command > output.txt
```

Přesměrování standardního výstupu programu my_command do souboru **output.txt**. (Soubor output.txt je vytvořen. Pokud již existuje, je výstup programu my_command **připojen** na jeho konec.)

```
$ my_command >> output.txt
```

Podobná pravidla platí pro standardní **chybový** výstup, v tomto případě se používají následující operátory:

```
$ my_command 2> errors.txt
$ my_command 2>> errors.txt
```

Spojování výstupních proudů

Standardní výstup **a** standardní chybový výstup programu my_command lze současně přesměrovat do souboru **output.txt**.

```
$ my_command &> output.txt
```

```
$ my_command &>> output.txt
```

funguje v nových verzích bash

Alternativní řešení pro &>>: Nejdříve je nutné **přesměrovat** standardní výstup a poté **spojit** standardní chybový výstup s výstupem standardním.

```
$ my_command >> output.txt 2>&1
```

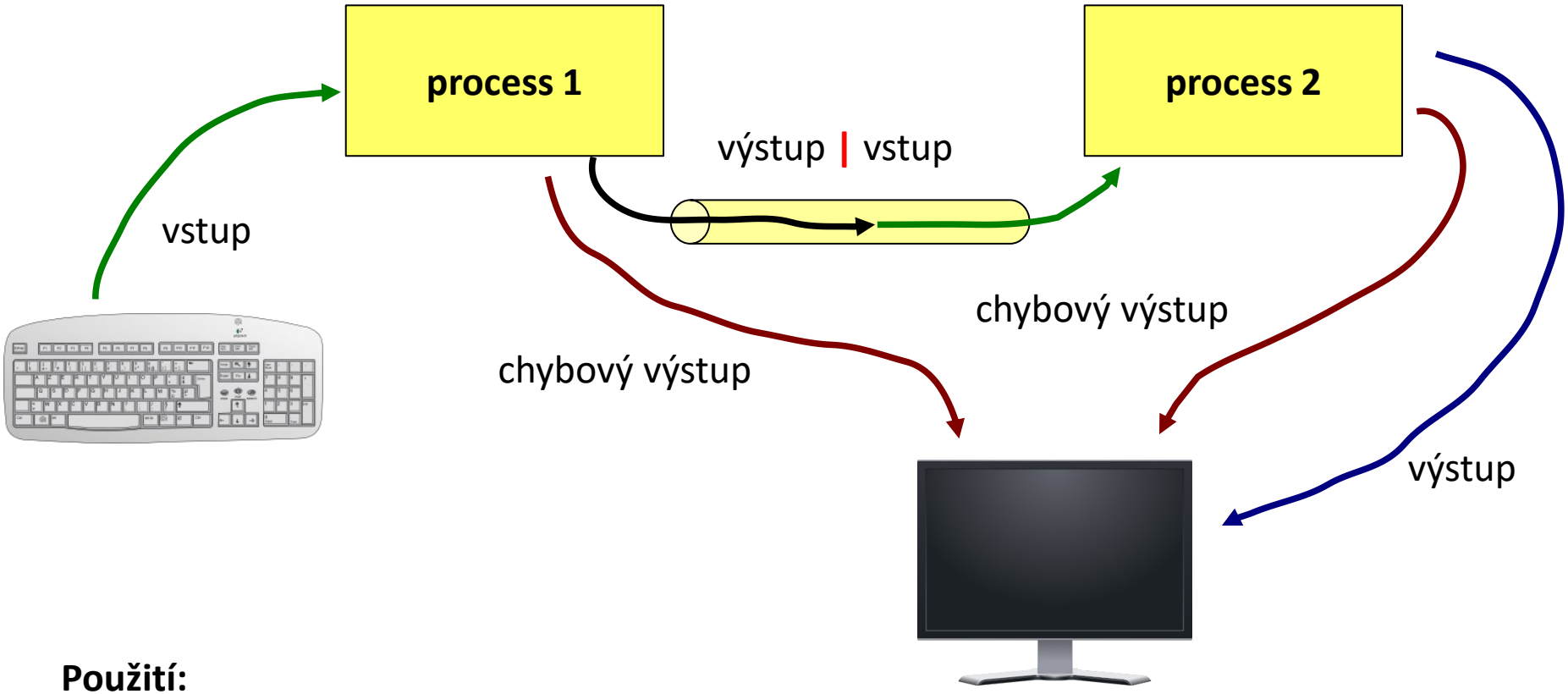
pořadí je důležité!

~~`$ my_command 2>&1 >> output.txt`~~

nefunguje

Roury (pípy)

Roury slouží ke spojování standardního výstupu jednoho procesu se standardním vstupem jiného procesu.

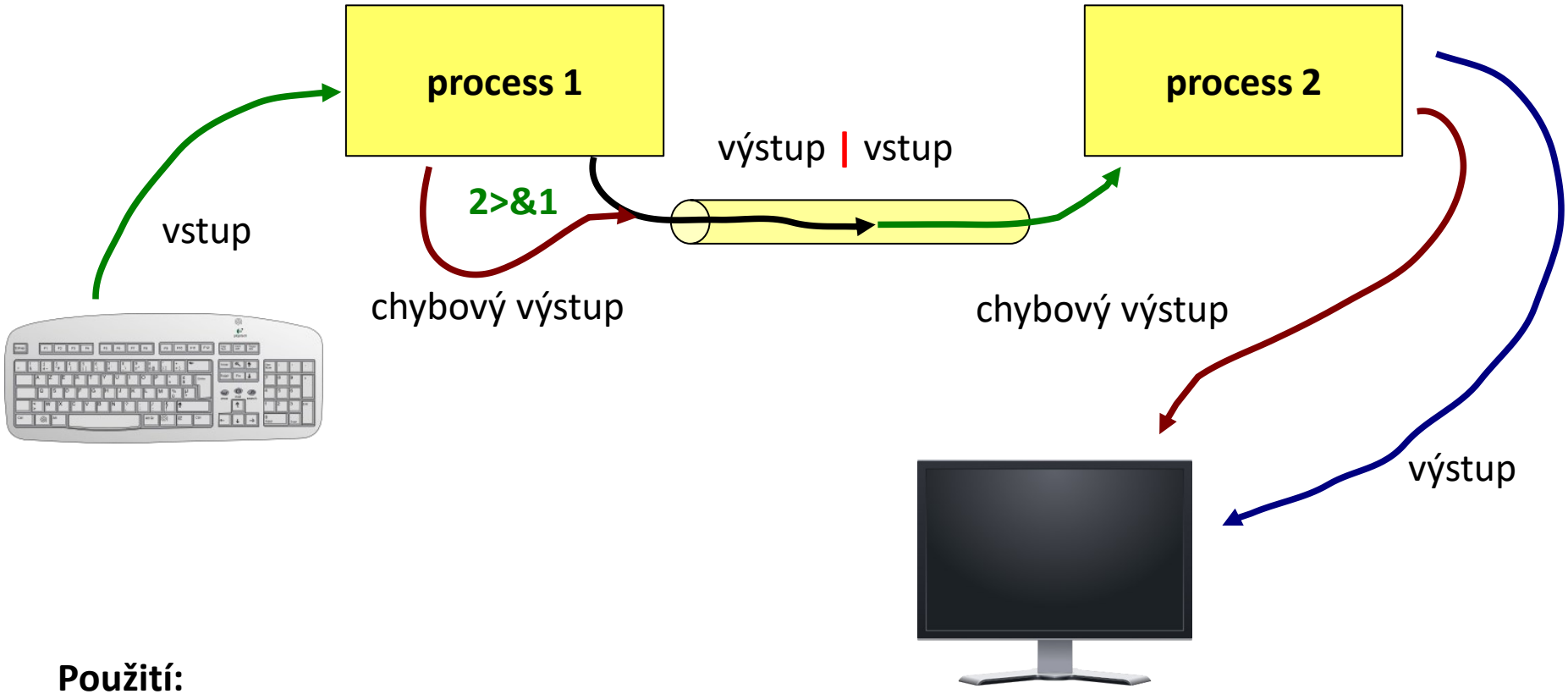


Použití:

```
$ command_1 | command_2
```

Roury a chybový proud

Přenos standardního chybového výstupu přes rouru je možné provést po jeho spojení se standardním výstupem.



Použití:

```
$ command_1 2>&1 | command_2
```

Příkazy pro cvičení

cat	spojí obsah více souborů do jednoho (za sebe), případně vypíše obsah jednoho souboru
paste	spojí obsah více souborů do jednoho (vedle sebe)
wc	informace o souboru (počet řádků, slov a znaků)
head	vypíše úvodní část souboru
tail	vypíše koncovou část souboru

Ukázky použití:

```
$ cat soubor1.txt soubor2.txt
```
spojí obsah souborů soubor1.txt a soubor2.txt za sebe a výsledek vypíše na obrazovku

```
$ paste soubor1.txt soubor2.txt
```
spojí obsah souborů soubor1.txt a soubor2.txt vedle sebe a výsledek vypíše na obrazovku

```
$ wc soubor.txt
```
vypíše počet řádků, slov a znaků, které obsahuje soubor soubor.txt

```
$ head -15 soubor.txt
```
vypíše prvních 15 řádků ze souboru soubor.txt

```
$ tail -6 soubor.txt
```
vypíše posledních 6 řádků ze souboru soubor.txt

Příkazy pro cvičení ...

Příkaz **tr** slouží k transformaci nebo mazání znaků ze standardního vstupu. Výsledek je zasílán do standardního výstupu.

Příklady:

```
$ cat soubor.txt | tr --delete "qwe"
```

z obsahu souboru **soubor.txt** odstraní znaky "q", "w" a "e"

```
$ cat soubor.txt | tr --delete "[:space:]"
```

z obsahu souboru **soubor.txt** odstraní všechny bílé znaky

```
$ echo $PATH | tr ":" "\n"
```

v textu zaslaného příkazem echo budou nahrazeny znaky ":" znakem nového řádku "\n"

Cvičení 1

1. Nalezněte všechny soubory s koncovkou **.f90**, které obsahuje adresář **/home/kulhanek/Documents/C2110/Lesson03**. Seznam souborů uložte do souboru **~/Procesy/seznam.txt**
2. Kolik řádků obsahuje soubor **seznam.txt**?
3. Vypište první dva řádky ze souboru **seznam.txt** nejdříve na obrazovku a poté do souboru **dva_radky.txt**
4. Vypište pouze třetí řádek ze souboru **seznam.txt**
5. V adresáři **/proc** nalezněte všechny soubory, které začínají písmeny **cpu**. Z výpisu odstraňte informace o nepovoleném přístupu přesměrováním chybového proudu do **/dev/null**
6. Vypište názvy adresářů obsažené v proměnné **PATH**, každý na jeden řádek.