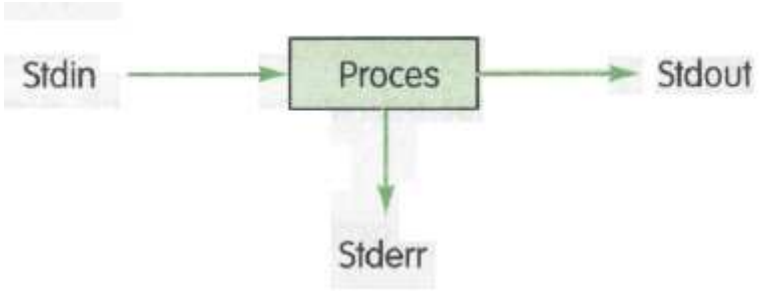
Przekierowanie strumienia danych

Każdy uruchomiony w Linuksie proces pobiera skądś dane, gdzieś wysyła wyniki swojego działania i komunikaty o błędach. Dane przesyłane są między urządzeniami w postaci strumieni. Strumienie danych przypisane każdemu procesowi są pokazane na (rys. 6.6).

- stdin (standard input), czyli standardowe wejście, z którego proces pobiera dane (domyślnie jest to klawiatura),
- stdout (standard otuput) to standardowe wyjście, z którego wysyłany jest wynik działania procesu (domyślnie jest to ekran),
- stderr (standard error) to standardowe wyjście błędów, gdzie trafiają wszystkie komunikaty o błędach (domyślnie ekran).



Rys. 6.6. Strumienie danych

Linux wszystkie urządzenia traktuje jak pliki, niezależnie od tego, czy to jest plik, folder, urządzenie blokowe (klawiatura, ekran), czy strumień. Powłoka Linuksa identyfikuje je za pomocą przyporządkowanych im liczb całkowitych, tak zwanych **deskryptorów plików**:

0 to plik, z którego proces pobiera dane (stdin),
1 to plik, do którego proces zapisuje wyniki swojego działania (stdout),
2 to plik, do którego trafiają komunikaty o błędach (stderr),

Za pomocą operatorów przypisania można manipulować strumieniami, poprzez przypisanie deskryptorów: 0,1,2 innym plikom, niż tym, które reprezentują klawiaturę i ekran.

Standardowe wejście i wyjście (strumienie danych) możemy przekierować. Do tego celu przygotowano trzy operatory:

- znak „<” umożliwia przekierowanie zawartości pliku do standardowego wyjścia, np. **more < plik**,
- znak „>” umożliwia przekierowanie strumienia danych ze standardowego wyjścia do pliku; jeżeli plik istnieje, to jego poprzednia zawartość zostaje usunięta, np. **ls > plik**,
- znaki „>>” umożliwiają przekierowanie strumienia danych ze standardowego wyjścia do pliku; jeżeli plik istnieje, to nowe dane zostają dopisane na końcu pliku.

Przełączanie standardowego wyjścia

Wynik jakiegoś polecenia można wysłać do pliku, a nie na ekran. Do tego celu używa się operatora:

> plik

Przykład: **ls -la /usr/bin > ~/wynik**

Rezultat działania polecenia **ls -la /usr/bin** trafi do pliku o nazwie **wynik** jeśli wcześniej nie istniał plik o takiej samej nazwie, to zostanie utworzony, jeśli istniał, cała jego poprzednia zawartość zostanie nadpisana.

Jeśli chcemy, aby dane wyjściowe dopisywane były na końcu pliku, bez wymazywania jego wcześniejszej zawartości, stosujemy operator:

>> plik

Przykład: **free -m >> ~/wynik**

Wynik polecenia **free -m** (pokazuje wykorzystanie pamięci RAM i swap) zostanie dopisany na końcu pliku **wynik**, nie naruszając jego wcześniejszej zawartości.

Potokowanie strumienia danych

Zastosowanie znaku | pozwala na łączenie wyjścia jednego polecenia z wejściem innego. Dane wygenerowane za pomocą pierwszego polecenia, przekazane zostaną na wejście następnego polecenia i po przetworzeniu przekazane na wejście kolejnego lub na ekran. Tego typu przetwarzanie danych nazywane jest **potokiem**. Polecenia często wykorzystywane w potokach:

1. **more** - służy do przeglądania tekstu strona po stronie, jeden ekran na raz, przewijanie stron możliwe tylko „do przodu”, np. ls -la | more,
2. **less** - podobnie jak more, ale przewijanie stron możliwe w obu kierunkach, np. ls -la | less,
3. **cat** - polecenie wyświetla na ekranie zawartość pliku tekstowego, np. cat /etc/passwd | less,
4. **grep** - przeszukuje wskazany strumień danych, szukając linii zawierających ciąg znaków pasujących do podanego wzorca, cat /etc/passwd | grep uczen
5. **wc** - wypisuje liczbę bajtów, słów lub linii w plikach, np. ls -la | wc -1.

Przykład potoku: ls -la | grep plik | wc –l