raises the *overflow* or *underflow* exception. (Implementations that conform to the IEEE standard will have this property.) `feraiseexcept` returns zero if `excepts` is zero or if all specified exceptions were successfully raised; otherwise, it returns a nonzero value.

*fesetexceptflag*

The `fesetexceptflag` function attempts to set the floating-point status flags represented by `excepts`. The states of the flags are stored in the `fexcept_t` object pointed to by `flagp`; this object must have been set by a previous call of `fegetexceptflag`. Moreover, the second argument in the prior call of `fegetexceptflag` must have included all floating-point exceptions represented by `excepts`. The `fesetexceptflag` function returns zero if `excepts` is zero or if all specified exceptions were successfully set; otherwise, it returns a nonzero value.

*fetestexcept*

The `fetestexcept` function tests only those floating-point status flags represented by `excepts`. It returns the bitwise *or* of the floating-point exception macros corresponding to the flags that are currently set. For example, if the value of `excepts` is `FE_INVALID | FE_OVERFLOW | FE_UNDERFLOW`, the `fetestexcept` function might return `FE_INVALID | FE_UNDERFLOW`, indicating that, of the exceptions represented by `FE_INVALID`, `FE_OVERFLOW`, and `FE_UNDERFLOW`, only the flags for `FE_INVALID` and `FE_UNDERFLOW` are currently set.

## Rounding Functions

```
int fegetround(void);
int fesetround(int round);
```

The `fegetround` and `fesetround` functions are used to determine the rounding direction and modify it. Both functions rely on the rounding-direction macros (the third group in Table 27.8).

*fegetround*

The `fegetround` function returns the value of the rounding-direction macro that matches the current rounding direction. If the current rounding direction can't be determined or doesn't match any rounding-direction macro, `fegetround` returns a negative number.

*fesetround*

When passed the value of a rounding-direction macro, the `fesetround` function attempts to establish the corresponding rounding direction. If the call is successful, `fesetround` returns zero; otherwise, it returns a nonzero value.

## Environment Functions

```
int fegetenv(fenv_t *envp);
int feholdexcept(fenv_t *envp);
int fesetenv(const fenv_t *envp);
int feupdateenv(const fenv_t *envp);
```