Wojciech Myszka

Laboratorium 11: Struktury danych: Operacje z uwzględnieniem błędów

2016-05-07 09:06:00 +0200

1. Wprowadzenie

1.1. Liczby przybliżone

Na zajęciach z Technologii Informacyjnych pojawił się problem arytmetyki liczb obarczonych błędem bezwzględnym.

Liczba przybliżona według używanej definicji to para, na którą składa się wartośc przybliżóna (a) i jej błąd bezwzględny (Δa).

Aby móc korzstać z takich „tworów" w języku C można użyć struktur. Tworzymy nowy typ danych (real) będący strukturą o dwu składowych: val (wartość przybliżona) i err (błąd bezwzględny).

Zatem definiujemy **struct** real jako:

```
struct real {
  double val;
  double err;
}
struct real x;  // a tu deklarujemy zmienną takiego typu
```

x.val przechowuje wartość liczby, x.err przechowuje jej **błąd bezwzględny**.

Strukture **struct** real powinna być zadeklarowana jako struktura globalna (to znaczy na zewnątrz funkcji main()).

Ponieważ na zmiennych typu nie można wykonywać bezpośrednio żadnych operacji — należy przygotować odpowiednie funkcje.

Należy stworzyć **bibliotekę** (zestaw funkcji) operacji arytmetycznych (co najmniej suma, różnica, iloczyn, iloraz) na takich liczbach, która oprócz wykonywania samych działań, będzie za każdym razem wyliczała błąd bezwzględny wyniku.

Prototyp funkcji suma może wyglądać tak:

```
struct real suma(struct real a, struct real b);
```

A odpowiednikiem prostego działania

$$y = \frac{a+b}{cd}$$

będzie

```
y = iloraz(suma(a, b), iloczyn(c, d));
```

1.2. Zaokrąglanie

Biblioteka powinna też być wyposażona w funkcje **poprawnego** zaokrąglania wyniku (działającą dla dowolnych liczb: zaokrąglenie z dokładnością ...,tysięcy, setek, dziesiątek, jedności, dziesiątych, setnych, tysięcznych...). Podczas zaokrąglania powinien być odpowiednio korygowany błąd bezwzględny. Prototyp funkcji zaokrąglania wyglądać może jakoś tak:

```
struct real zaokr(struct real wartosc, int n);
```

gdzie wartosc to wartość przybliżona, którą chcemy zaokrąglić, a n to liczba cyfr znaczących, która nas interesuje. Dla n = 0 — chcemy tylko wartości całkowitej (dokładność jednostek, 10^0), n = 1 — dokładność do dziesiątek (10^1), a n = -2 to dwie cyfry po kropce dziesiętnej (dokładność do setnych części 10^{-2}). Stusujemy regułę „zaokrąglania poprawnego" (jeżeli odrzucana cyfra jest z zakresu 0 do 4 to pozostająca cyfra jest bez zmian, w przeciwnym razie pozostającą cyfrę zwiększamy o 1).

Zaokrąglenie do pełnych jednostek można zrealizować w sposób następujący. Jeżeli x to dowolna liczba to **część_całkowita**$(x + 0.5)$ da nam w rezultacie poprawnie zaokrągloną (do jednostek) wartość.

Do wyznaczania części całkowitej służą funkcje trunc, ceil, floor, round,...

2. Polecenie typedef

Ponieważ ciągłe pisanie **struct** real może wydawać się męczące, można sobie ułatwić życie korzystając z polecenia **typedef**, które pozwala stworzyć nowy typ (na przykład strukturalny):

```
typedef struct{
    double val;
    double err;
} real;
```

tworzy nowy typ (o nazwie **real**) równoważny używanemu wcześniej **struct** real.

3. Zadania do wykonania

1. Biblioteka funkcji arytmetycznych.
2. **Przetestowana** funkcja realizująca operację zaokrąglenia.
3. Powtórzenia obliczeń zaprezentowanych na slajdach wykładu (Przykład) (lub, jeżeli to za trudne, jakieś inne „sensowne” obliczenia — objętość jakiejś popularnej bryły geometrycznej, rozwiązanie równania kwadratowego — potrzebne będzie oszacowanie błędu pierwiastka,...).

4. Wersja PDF tego dokumentu...

...pod adresem.

Wersja: 50 z drobnymi modyfikacjami! data ostatniej modyfikacji 2016-05-07 09:06:00 +0200