2. *Данаев Н.Т., Смагулов Ш.* Об одной методике численного решения уравнения Навье-Стокса в переменные (ψ,ω). Численные методы механики сплошных сред // РЖ. — Серия ВМ. — № 2. — С. 27–35.
3. *Смагулов Ш.* Математические вопросы приближенных методов для уравнений Навье-Стокса: Дис. … д-ра физ.-мат. наук. — Новосибирск, 1988. — 380 с.
4. *Алибиев Д.Б.* Об одном методе приближенного решения уравнения Навье-Стокса // Математическая кибернетика и управления движениями. — Алматы, 1991. — С. 39–44.

ӘОЖ 37.01:378.096:004

Д.Р.Бейсенова, Р.Ж.Төлеуханова

Е.А.Бөкетов атындағы Қарағанды мемлекеттік университеті

ДЕМОНСТРАЦИЯЛЫҚ МЫСАЛДАР ӘДІСІ НЕГІЗІНДЕ ПРОГРАММАЛАУДЫ ОҚЫТУ

В статье рассматривается метод демонстрационных примеров в обучении программированию на языке Си. На примере операций сложения, вычитания, логических, условных операций и выражений выявлены преимущества данного метода.

There is method of demonstrational example in the learning to program by C language is considered in this article. It shown this method is worth of operating addition, substraction, logical and conditional operation.

Ақпараттық оқу модельдері — демонстрациялық мысалдар — модельдердің жаңа класы, оларда дәстүрлі оқу және компьютерлік оқыту модельдерінің қасиеттері бар. Демонстрациялық мысалдар әдісін негізінде тек қана программалауды оқыту барысында ғана емес, информатиканың басқа бөлімдерін оқыту кезінде де қолдануға болады.

Демонстрациялық мысалдар әдісін қолдану студенттерді программаны оқуға үйретуге мүмкіндік береді. Басқа адамдар жазған программаларды оқу дағдысы программаның дұрыстығын анықтаудың, программаның прагматикалық аспектісін табу, оларды модификациялау және өзіндік программаларды верификациялау үшін негіз болып табылады. Жеке дидактикалық әдістердің әр түрлілігі оқытудың арнайы әдістері деп аталады, олар нақты пәндік облыстың мәнін анық көрсетеді.

Демонстрациялық мысалдар әдісі — бұл демонстрациялық мысалдар (компьютерлік оқу модельдері) қолдануға негізделген оқыту әдісі. «Компьютерлік оқу модельдері» түсінігі, В.В.Лаптев және М.В.Швецкийлердің анықтауынша, ақпараттық модель негізінде біріктірілген құбылыс пен зерттеудің оқыту объектісінің қарым-қатынасынан тұратын программалық орта [1].

Си тілінің операцияларын түсіндіруді демонстрациялық мысалдар негізінде келесі түрде көрсетуге болады.

Қосу және азайту операциялары

Тілде қосу мен азайтудың дәстүрлі емес екі операциясы бар, олар ++ және – – деп белгіленеді. ++ операциясы операндаға 1-ді қосады, ал – – 1-ді алады. Бұл операциялар өз операндаларының алдында немесе соңында қолданылуы мүмкін. Олар өрнекте әр түрлі амал жасайды: ++n жазуы n мәнін қолданбай тұрғанда үлкейтеді, ал n++ жазуында n-нің мәні қолданылғаннан кейін үлкейеді. Егер n=5 болса, m=++n операторы орындалған соң m-нің мәні 6-ға, ал m=n++ орындалған соң m=5. Екі жағдайда да n=6. Бірінші мысал n=n+1; m=n тізбегіне сәйкес, ал екінші жағдайда: m=n; n=n+1 тізбегіне сай. Азайту операциясы тура солай. Екі операцияның да басымдылығы арифметикалық операциялар арасында ең кішісі. 1 программа осы командаларды қолдануды көрсетеді.

Разрядты логикалық операциялар бүтін сан немесе таңбаның бөлек биттерімен жұмыс істеу үшін арналған:

& — разрядты ЖӘНЕ
^ — НЕМЕСЕ разрядты болдырмау
>> — оңға жылжу
| — разрядты НЕМЕСЕ
<< — солға жылжу

```
~ — инверсия
/* өз цифрларының қосындысына
бөлінетін екі таңбалы сандары анықтау */
# incude <stdio.h>
main()
{
  int a,b,k,s,c;
  k=0; a=1;
  while (a <= 9)
  {
    b=0;
    while (b <= 9)
    {
      s=a+b;
      c=a*10+b;
      if (c%s == 0)
      {
    printf("%d ",c);
        k++;
      }
    b++;
    }
    a++; printf("\n");
  }
printf("Барлығы:%d\n",k);
}
```

1-программа

Разрядты ЖӘНЕ (&) операциясы екілік разрядтардың кейбір топтарын бөліп алу үшін қолданылады: мысалы, *n* = *n* & 0177 жазуы *n* санының барлық екілік разрядтарын 0-ге қояды (кішілерінен басқасын). НЕМЕСЕ (|) операциясы бөлек разрядтарды бірге қою үшін қолданылады. Мысалы, *m* = *m* | OxFOF; жазуы m санының разрядтарын 11, 10, 9, 8, 3, 2, 1 және 0-ді «қосады». << және >> операциялары берілген разряд санына операндты солға немесе оңға жылжытады. Мысалы, *m* << 3*m* мәнін солға қарай 3 разрядқа жылжытады, босаған кіші разрядтарды нөлмен толтыра отырады. Бұл санды 8-ге көбейткенге сәйкес (жалпы жағдайда *n* дәрежесі — 2, бұл жерде *n* — жылжитын разрядтар саны). Оңға қарай жылжу 2*n* бөлумен тең. Бұл жылжуда босаған разрядтар таңбалы биттің мәнімен толтырылады. Унарлы операция санның (таңбаның) екілік разрядтарын инверсиялауды орындайды, яғни әрбір бірлік битті нөлге айналдырады және қарама-қарсы [2].

Кейбір разрядты логикалық операцияларды қолдануды мысалдарда көрсетеміз. 2-программадағы берілген бүтін саны оналтылық түрде:

m = OXIFF; деп алынған. Нәтижені шығару үшін PRINT функциясы қарастырылған. Нәтижені сараптауды ыңғайландыру үшін сандарды оналтылық, сегіздік және ондық түрлерде баспаға береді, ол үшін x, o, d спецификацияларын қолданамыз.

```
/* разрядты логикалық операциялар */
#include <stdio.h>
PRINT(n)
int n;
{
    printf("%5x %5o %5d\n",n,n,n);
}
 main()
{
    int m,n;
    m=0X1F3; PRINT(m); /*16-лық сан*/
    n=m&0177; PRINT(n); /*7 кіші битті бөліп алу*/
    n=m|013; PRINT(n); /*4 кіші битті қою */
```

```
    n=m>>4; PRINT(n); /*оңға жылжу*/
    m=n<<3; PRINT(m); /*солға жылжу*/
}
```

2-программа

Келесі программа (3-программа) санды оңға қарай бір битке жылжыту командасын (m=m>>1;) және санның кіші разрядын бөліп алуды (m&01) қолданып, берілген FOF санының бірлік биттерінің санын есептейді де, нәтижені баспаға береді.

```
/* бірліктерді санау */
#include <stdio.h>
main()
{
  int m,k;
  k=0;
  m=0xf0f;
  while (m !=0)
  {
    if (m & 01)
        k++;
    m=m >> 1;
  }
  printf("k=%d\n",k);
}
```

3-программа

Меншіктеу операциялары және өрнектер

Сол жақ бөлігі оң жақ бөлікте қайталанса, онда i=i + b өрнегін қысқа түрде былай жазамыз: i+=b.

Бинарлы операцияларды op= түрлі жазуға болады, мұнда *op* – мына + -* / % |^ & << >> операцияларының бірі. Егер E1 және E2 — өрнектер болса, онда E1 op = E2 жазуына E1 = (E1) op (E2) пара-пар. E2-нің жақшаларына көңіл аударсақ: x*=y+1 меншіктелуі негізі x=x*(y+1) білдіреді (4-программа). Осы айтылғандарды келесі программамен көрсетейік.

Берілген n-натурал саны арқылы m саны құрылады. m саны n санының цифрларымен, кері қарай алынған ретпен жазылады. m*=10; m+=z жазуына қарағанда m=m*10+z жазуы әдемі көрінеді.

Шартты операция

Негізі ол if – then – else операторының қысқартылған түрі, жалпы алғанда былай жазылады: өрнек1: өрнек2: өрнек3. Егер «өрнек1» нөлге тең болмаса, операция нәтижесі «өрнек2» мәніне, қарсы жағдайда «өрнек3» мәніне тең. Шартты операция тернарлы деп аталады, оны меншіктеу операторында қолдануға болады.

```
main()
{
    int n,z,m=0;
    printf(“енгіз n\n");
    scanf("%d",&n);
    while (n !=0)
    {
      z=n%10;
      n/=10;
      m*=10;
      m+=z;
    }
    printf("m=%d\n",m);
}
```

4-программа

Сонымен, if (x > y) max=x; else max=y; орнына max = (x > y) ? x : y деп жазуға болады. «өрнек1»-дің жанына жақша қою міндетті емес, өйткені ?: басымдылығы өте төмен, ол тек меншіктеуден төмен. Шартты операциясы қысқа программаларды жазуға мүмкіндік береді. 1-ден m-ге дейінгі натурал сандардың квадраттарын баспаға беретін циклді программа құрайық, әр жолда 6 саннан; әр сан 5 орын алады және бір баған мен бағанның арасында 1 бос орын болады, әр жол келесі жолға өту (\n) таңбасымен аяқталады. Жаңа жолға көшу әрбір алты элементтен соң және m-нан кейін жүргізіледі [3].

```
/* натурал сандардың квадраттары */
#include <stdio.h>
main()
{
  int m,i=1;scanf("%d",&m);
  while (i <= m)
  {
      printf("%5d%c",i*i, (i%6==0| |i==m)? '\n':");
      i++
  }
}
```

5-программа

Сонымен, демонстрациялық мысалдар әдісін программалауды және информатиканың басқа бөлімдерін оқыту кезінде қолдануға болады.

Демонстрациялық мысалдар әдісін программалауды оқыту барысында төмендегідей артықшылықтары бар:

1. Демонстрациялық мысалдар әдісін қолдану студенттерді программаны оқуға үйретуге мүмкіндік береді. Басқа адамдар жазған программаларды оқу дағдысы программаның дұрыстығын анықтаудың программаның прагматикалық аспектісін табу, оларды модификациялау және өзіндік программаларды верификациялау үшін негіз болып табылады.

2. Бұл әдіс демонстрациялық мысалдарды басқа оқу пәндерінде кездесетін есептерді программалау кезінде және де курстық және дипломдық жұмыстарды жазу кезінде пайдалануға мүмкіндік береді. Оқытуда, демонстрациялық мысалдарды қолдану тәжірибесі көрсеткендей, студенттер өздерінің программалық жобаларын жасауда көбінесе жиналған және нақты программалауды қолданады. Синтездейтін программалауды студенттер әдістемелік зерттеу жұмыстары кезінде пайдаланады (мысалы, жаңа тақырыпқа демонстрациялық мысалдар жинаған кезде).

3. Оқытушы демонстрациялық мысалдарды студенттерде біліктілік пен дағды қалыптастыру үшін ғана емес, жаңа теориялық материалды оқыту кезінде де қолдана алады.

4. Демонстрациялық мысалдар әдісін қолдану студенттер мен оқытушының қарым-қатынасын күшейтеді, басқа студенттер жасаған демонстрациялық мысалдармен алмасуға және оларды модификациялауға мүмкіндік береді.

Әдебиеттер тізімі

1. *Косова И.С.* Использование языка LISP при обучении функциональному программированию будущих учителей математики и информатики: Автореф. дис. … канд. пед. наук. — СПб., 2001. — 19 с.
2. *Брайан У. Керниган, Деннис М. Ритчи.* Язык программирования С. — 2-е изд. / Пер. с англ. – М.: Изд. дом «Вильямс», 2006. — 304 с.
3. *Макогон В.С.* Язык программирования Си для начинающих: Учеб. пособие. — Одесса: Астропринт, 1993. — 96 с.