Функции II

Калин Георгиев

29 ноември 2018 г.

Отново функции vs. подпрограми

Математически изображения

- Приличат на “Формули”: $S = vt + \frac{1}{2}at^2$
- Съотвена функция: $S : \mathcal{R} \times \mathcal{R} \times \mathcal{R} \to \mathcal{R}$, $S(v,t,a) = vt + \frac{1}{2}at^2$
- Могат да участват в изрази: $S(10,60,0) + S(10,60,20)$
- Не “правят” нищо

```
double displacement (double speed,
                     double time,
                    double acceleration)
{
  double S = speed*time + acceleration*time*time/2;
  return S;
}
void main ()
{
  //....
  cout << displacement (10,60,0) + displacement (10,60,20);
}
```

Математически изображения

- Приличат на “Формули”: $S = vt + \frac{1}{2}at^2$
- Сьотвена функция: $S : \mathcal{R} \times \mathcal{R} \times \mathcal{R} \to \mathcal{R}$, $S(v, t, a) = vt + \frac{1}{2}at^2$
- Могат да участват в изрази: $S(10, 60, 0) + S(10, 60, 20)$
- Не “правят” нищо

```
double displacement (double speed,
                     double time,
                    double acceleration)
{
  double S = speed*time + acceleration*time*time/2;
  return S;
}
void main ()
{
  //....
  cout << displacement (10,60,0) + displacement (10,60,20);
}
```

Математически изображения

- Приличат на “Формули”: $S = vt + \frac{1}{2}at^2$
- Съотвена функция: $S : \mathcal{R} \times \mathcal{R} \times \mathcal{R} \rightarrow \mathcal{R}$, $S(v, t, a) = vt + \frac{1}{2}at^2$
- Могат да участват в изрази: $S(10, 60, 0) + S(10, 60, 20)$
- Не “правят” нищо

```
double displacement (double speed,
                     double time,
                    double acceleration)
{
  double S = speed*time + acceleration*time*time/2;
  return S;
}
void main ()
{
  //....
  cout << displacement (10,60,0) + displacement (10,60,20);
}
```

Математически изображения

- Приличат на “Формули”: $S = vt + \frac{1}{2}at^2$
- Сътовена функция: $S : \mathcal{R} \times \mathcal{R} \times \mathcal{R} \to \mathcal{R}$, $S(v, t, a) = vt + \frac{1}{2}at^2$
- Могат да участват в изрази: $S(10, 60, 0) + S(10, 60, 20)$
- Не “правят” нищо

```
double displacement (double speed,
                     double time,
                    double acceleration)
{
  double S = speed*time + acceleration*time*time/2;
  return S;
}
void main ()
{
  //....
  cout << displacement (10,60,0) + displacement (10,60,20);
}
```

Математически изображения

- Приличат на “Формули”: $S = vt + \frac{1}{2}at^2$
- Съотвена функция: $S : \mathcal{R} \times \mathcal{R} \times \mathcal{R} \to \mathcal{R}$, $S(v,t,a) = vt + \frac{1}{2}at^2$
- Могат да участват в изрази: $S(10,60,0) + S(10,60,20)$
- Не “правят” нищо

```
double displacement (double speed,
                     double time,
                    double acceleration)
{
  double S = speed*time + acceleration*time*time/2;
  return S;
}
void main ()
{
  //....
  cout << displacement (10,60,0) + displacement (10,60,20);
}
```

Подпрограми. Процедури

- “Правят” нещо: Страничен ефект
- “Стойността” им няма значение

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start; element <= end; element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{
  pritnSequence (1,10,1);
  pritnSequence (10,30,2);
  pritnSequence (30,80,5);
}
```

Процес на изпълнение. Програмен стек

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(1)

step	1

start	1
end	10
step	1

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(1)

step	1

start	1
end	10
step	1

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(2)

step	2

start	10
end	30
step	2

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(2)

step	2

start	10
end	30
step	2

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(3)

step	15

start	30
end	80
step	5

Формални vs. Фактически параметри

```
void pritnSequence (long start, long end, long step)
{
  for (long element = start;
       element <= end;
       element += step)
  {
    cout << element;
    if (element < end)
      cout << ",";
  }
  cout << endl;
}

void main ()
{ long step = 1;
  pritnSequence (1,10,step); //(1)
  step = 2;
  pritnSequence (10,30,step); //(2)
  step = 15;
  pritnSequence (30,80,5); //(3)
}
```

(3)

step	15

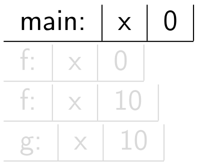
start	30
end	80
step	5

Взаимни извиквания

```
void g (long x)
{cout << x;}

void f (long x)
{
  x = x + 10;
  g (x);
}

void main ()
{
  long x = 0;
  f (x);
  cout << x;
}
```

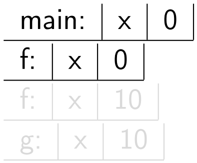


Взаимни извиквания

```
void g (long x)
{cout << x;}

void f (long x)
{
  x = x + 10;
  g (x);
}

void main ()
{
  long x = 0;
  f (x);
  cout << x;
}
```

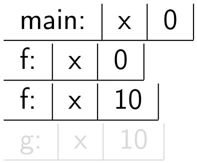


Взаимни извиквания

```
void g (long x)
{cout << x;}

void f (long x)
{
  x = x + 10;
  g (x);
}

void main ()
{
  long x = 0;
  f (x);
  cout << x;
}
```

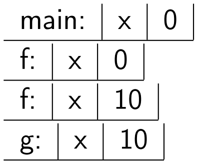


Взаимни извиквания

```
void g (long x)
{cout << x;}

void f (long x)
{
  x = x + 10;
  g (x);
}

void main ()
{
  long x = 0;
  f (x);
  cout << x;
}
```



Взаимни извиквания

main:	x	0

```
void g (long x)
{cout << x;}

void f (long x)
{
  x = x + 10;
  g (x);
}

void main ()
{
  long x = 0;
  f (x);
  cout << x;
}
```

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
    printSequence (N-1);
 }
  cout << N << "␣";
}
void main ()
{
   printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
     printSequence (N-1);
 }
   cout << N << "␣";
}
void main ()
{
    printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
    printSequence (N-1);
 }
  cout << N << "␣";
}
void main ()
{
   printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
    printSequence (N-1);
 }
  cout << N << "␣";
}
void main ()
{
   printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
    printSequence (N-1);
 }
  cout << N << "␣";
}
void main ()
{
   printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Самоизвиквания

```
void printSequence (long N)
{
 if (N > 0)
 {
    printSequence (N-1);
 }
  cout << N << "␣";
}
void main ()
{
   printSequence(4);
}
```

N	4
N	3
N	2
N	1
N	0

Размяна

```
void printSequence (long N)
{
  cout << N << "␣";
  if (N > 0)
  {
    printSequence (N-1);
  }
}
void main ()
{
  printSequence(4);
}
```

Пример:

- Въвеждане на число във фиксиран интервал

```
void main ()
{
  cout << enterNumber (0,100) / enterNumber (1,100);
}
```

- Отпечатване на цифри

Благодаря за вниманието!