Семинар 4. Язык С. Основные конструкции языка.

1. Пример использования базовых конструкций

```
/*============================================================*/
/* Программа "salary.c"                                       */
/*============================================================*/
#include <stdlib.h> // Заголовочный файл стандартной библиотеки С
#include <stdio.h>  // Заголовочный файл для функций ввода-вывода
#include <string.h> // Заголовочный файл для функций работы со строками символов
#include <errno.h>  // Заголовочный файл для функций расшифровки кодов ошибок
#include <math.h>   // Заголовочный файл для функций из математической библиотеки

// Определение нового типа:
typedef struct tag_MyStruct_t {
  char Name[20];
  int BirthYear;
  int BirthMonth;
  int BirthDay;
  double AverageSalary;
  double Salary[12];
} MyStruct_t;

// Определение глобальной константы:
#define Pi 3.1415926535897932384

// Макросы:
#define ABS(a)    ((a) > (0) ? (a) : (-a))
#define MIN(a,b)  ((a) < (b) ? (a) : (b))
#define MAX(a,b)  ((a) > (b) ? (a) : (b))

// Модифицированные стандартные функции:
double dsin(double x) {
  double s = sin(x);
  if (ABS(s)>1e-15)
    return s;
  else
  return 0.0;
}

// Головная программа:
int main() {
// Объявление переменных:
  MyStruct_t MyStruct;
  int i, j, n;
  double *aa, *bb, *cc;
  double **AA;
  double BB[10][10];

// Использование переменных:
  MyStruct.Name[11] = 0;
  strncpy(MyStruct.Name,"Petrov Ivan",11);

  MyStruct.BirthYear  = 1980;
  MyStruct.BirthMonth = 3;
  MyStruct.BirthDay   = 12;

  MyStruct.Salary[0]  = 30581.23;
  MyStruct.Salary[1]  = 30581.23;
  MyStruct.Salary[2]  = 30581.23 + 15000.00;
```

```
  MyStruct.Salary[3]  = 30581.23;
  MyStruct.Salary[4]  = 30581.23;
  MyStruct.Salary[5]  = 30581.23 + 23500.00;
  MyStruct.Salary[6]  = 30581.23;
  MyStruct.Salary[7]  = 30581.23;
  MyStruct.Salary[8]  = 30581.23 + 17800.00;
  MyStruct.Salary[9]  = 30581.23;
  MyStruct.Salary[10] = 30581.23;
  MyStruct.Salary[11] = 30581.23 + 30581.23 + 10000.00;

// Вычисления:
  MyStruct.AverageSalary = 0;
  for (i=0; i<12; i++) MyStruct.AverageSalary += MyStruct.Salary[i];
  MyStruct.AverageSalary /= 12;

// Вывод на экран:
  printf("Name=%s Birth=%02d.%02d.%04d AvSalary=%10.2lf\n",
    MyStruct.Name,
    MyStruct.BirthDay, MyStruct.BirthMonth, MyStruct.BirthYear,
    MyStruct.AverageSalary);

// Одномерные массивы переменной длины:
    n = 6; //11;
    aa = (double*)(malloc(sizeof(double)*n));
    bb = (double*)(malloc(sizeof(double)*n));
    cc = (double*)(malloc(sizeof(double)*n));

// Матрицы:
    AA = (double**)(malloc(sizeof(double*)*n));
    for(i=0; i<n; i++) {
      AA[i] = (double*)(malloc(sizeof(double)*n));
    }

// Заполнение массивов и вывод на экран:
    for(i=0; i<n; i++) {
      aa[i] = 0.2*i;
      bb[i] = pow(aa[i],3.5);
      cc[i] = dsin(Pi*aa[i]);
      printf("i=%2d aa=%lf bb=%le cc=%le\n",
             i,aa[i],bb[i],cc[i]);
    }

    for(i=0; i<n; i++) {
      printf("aa[%d]=%lf bb[%d]=%le cc[%d]=%le\n",
             i,aa[i],i,bb[i],i,cc[i]);
    }

// Заполнение матрицы и вывод на экран:
    for(j=0; j<n; j++)
      for(i=0; i<n; i++) {
        AA[i][j] = bb[i] + cc[j];
        printf("i=%2d j=%2d AA=%le\n",
               i,j,AA[i][j]);
      }

    printf("AA matrix:\n");
    for(j=0; j<n; j++) {
      for(i=0; i<n; i++) {
        AA[i][j] = bb[i] + cc[j];
```

```
        printf("%le ",AA[i][j]);
      }
      printf("\n");
    }

  return 0; // код возврата (ошибки)
}
```

2. Работа с массивами

Векторы, матрицы, тензоры

>cat massiv1.c

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
// Указатели на массивы
  double **A; // Матрица системы Ax=b
  double *B;  // Вектор правой части
  double *X;  // Вектор неизвестных
  double *C;  // Матрица системы в развернутом в строку виде
  int i, j, ij, n;
  char ch = 0;

  Met:
  printf("Input massiv dimension ");
  scanf ("%d",&n);
  printf("n=%d\n", n);

  // Выделение памяти для массивов
  A = (double**)(malloc(sizeof(double*)*n));
  for (i=0; i<n; i++)
    A[i] = (double*)(calloc(n, sizeof(double)));

  B = (double*)(malloc(sizeof(double)*n));
  X = (double*)(calloc(n, sizeof(double)));
  C = (double*)(calloc(n*n, sizeof(double)));

  // Заполнение элементов матрицы
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
      if (j==i) A[i][j] = 1.0*(i+1);
      else      A[i][j] = -0.1/(i+j);
      ij = i*n + j;
      C[ij] = A[i][j];
    }
  }

  // Заполнение элементов вектора правой части
  for (i = 0; i < n; i ++) {
    double el;
    printf("Input element with number %d = ", i);
    scanf("%lf",&el);
    B[i] = el;
  }

  // Вывод элементов матрицы
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
      printf ("A[%d][%d]=%lf\n", i, j, A[i][j]);
    }
  }
```

```
// Вывод элементов вектора правой части
  for (i = 0; i < n; i ++) {
    printf ("B[%d]=%lf\n", i, B[i]);
  }

// Решение системы
//...

// Освобождение выделенной памяти
  for (i=0; i<n; i++) free(A[i]);
  free(A);
  free(B);
  free(X);
  free(C);

  printf("\n");
  ch = getchar();
  printf("Continue?(y|n): ");

  ch = getchar();
  printf("\n");

  if (ch == 'y') {
    goto Met;
  }

  return 0;
}
```

>cat massiv2.c

```
#include <stdio.h>
#include <stdlib.h>

void print_system(int n, double **A, double *X);

int main ()
{
// Указатели на массивы
  double **A; // Матрица системы
  double *B;  // Вектор правой части
  double *X;  // Вектор неизвестных
  double *C;  // Матрица системы в развернутом в строку виде

  int i, j, k, ij, n, ier;
  double s, p;
  char ch = 0;

  Met:
  printf("Input massiv dimension ");
  scanf ("%d",&n);
  printf("n=%d\n", n);

  // Выделение памяти для массивов
  A = (double**)(malloc(sizeof(double*)*n));
  for (i=0; i<n; i++)
    A[i] = (double*)(calloc(sizeof(double),n));

  B = (double*)(malloc(sizeof(double)*n));

  X = (double*)(calloc(sizeof(double),n));
  C = (double*)(calloc(sizeof(double*),n*n));

  // Заполнение элементов матрицы
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
      if (j==i) A[i][j] = 1.0*(i+1);
      else      A[i][j] = -0.1/(i+j);
```

```
      ij = i*n + j;
      C[ij] = A[i][j];
    }
  }

  // Заполнение элементов вектора правой части
  for (i = 0; i < n; i ++) {
    double el;
    printf("Input element with number %d = ", i);
    scanf("%lf",&el);
    B[i] = el;
  }

  // Вывод элементов матрицы
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
      printf ("A[%d][%d]=%lf\n", i, j, A[i][j]);
    }
  }

  // Вывод элементов вектора правой части
  for (i = 0; i < n; i ++) {
    printf ("B[%d]=%lf\n", i, B[i]);
  }
  printf("\n");

  // Решение системы:
  ier =0;

  for (i=0; i<n; i++) X[i] = B[i];
  print_system(n,A,X);

  for (i=0; i<n; i++) {
    s = A[i][i];
    if (s!=0) {
      s = 1.0/s;
      A[i][i] = 1.0;
      for (j=i+1; j<n; j++) A[i][j] *= s;
      X[i] *= s;
      if (i<n-1){
        for (k=i+1; k<n; k++){
          p = A[k][i];
          A[k][i] = 0;
          for (j=i+1; j<n; j++) A[k][j] -= (p*A[i][j]);
          X[k] -= (p*X[i]);
        }
      }
      print_system(n,A,X);
    }
    else {
      ier = -1;
      printf("Bad matrix!\n");
      break;
    }
  }

  if (ier == 0){
    for (i=(n-2); i>=0; i--) {
      s = 0;
      for (j=i+1; j<n; j++) s += (A[i][j]*X[j]);
      X[i] -= s;
    }
    print_system(n,A,X);
  }

  // Проверка решения:
  if (ier == 0){
    for (i=0; i<n; i++) {
```

```
      s = B[i];
      for (j=0; j<n; j++) {
        ij = i*n + j;
      s -= (C[ij]*X[j]);
      }
      printf("i=%d s=%le\n",i,s);
    }
  }

  // Освобождение памяти:
  for (i=0; i<n; i++) free(A[i]);
  free(A);
  free(B);
  free(X);
  free(C);

  printf("\n");
  ch = getchar();
  printf("Continue?(y|n): ");

  ch = getchar();
  printf("\n");

  if (ch == 'y') {
    goto Met;
  }

  return 0;
}

void print_system(int n, double **A, double *X)
{
  int i, j;
  for (i=0; i<n; i++){
    for (j=0; j<n; j++) printf("%14.6le ",A[i][j]);
    printf("%14.6le\n",X[i]);
  }
  printf("\n");
  return;
}
```

>cat massiv3.c

```
#include <stdio.h>
#include <stdlib.h>

int solution_system(int n, double **A, double *X);

void print_system(int n, double **A, double *X);

int main ()
{
// Указатели на массивы
  double **A; // Матрица системы
  double *B;  // Вектор правой части
  double *X;  // Вектор неизвестных
  double *C;  // Матрица системы в развернутом в строку виде

  int i, j, k, ij, n, ier;
  double s;
  char ch = 0;

  Met:
  printf("Input massiv dimension ");
  scanf ("%d",&n);
  printf("n=%d\n", n);

  // Выделение памяти для массивов
```

```
A = (double**)(malloc(sizeof(double*)*n));
for (i=0; i<n; i++)
  A[i] = (double*)(calloc(sizeof(double),n));

B = (double*)(malloc(sizeof(double)*n));

X = (double*)(calloc(sizeof(double),n));
C = (double*)(calloc(sizeof(double*),n*n));

// Заполнение элементов матрицы
for (i=0; i<n; i++) {
  for (j=0; j<n; j++) {
    if (j==i) A[i][j] = 1.0*(i+1);
    else      A[i][j] = -0.1/(i+j);
    ij = i*n + j;
    C[ij] = A[i][j];
  }
}

// Заполнение элементов вектора правой части
for (i = 0; i < n; i ++) {
  double el;
  printf("Input element with number %d = ", i);
  scanf("%lf",&el);
  B[i] = el;
}

// Вывод элементов матрицы
for (i=0; i<n; i++) {
  for (j=0; j<n; j++) {
    printf ("A[%d][%d]=%lf\n", i, j, A[i][j]);
  }
}

// Вывод элементов вектора правой части
for (i = 0; i < n; i ++) {
  printf ("B[%d]=%lf\n", i, B[i]);
}
printf("\n");

// Подготовка к решению
for (i=0; i<n; i++) X[i] = B[i];

// Решение системы
ier = solution_system(n, A, X);

// Проверка решения
if (ier == 0){
  for (i=0; i<n; i++) {
    s = B[i];
    for (j=0; j<n; j++) {
      ij = i*n + j;
      s -= (C[ij]*X[j]);
    }
    printf("i=%d s=%le\n",i,s);
  }
}

// Освобождение памяти
for (i=0; i<n; i++) free(A[i]);
free(A);

free(B);
free(X);
free(C);

// Запрос на продолжение работы
printf("\n");
```

```
  ch = getchar();
  printf("Continue?(y|n): ");

  ch = getchar();
  printf("\n");

  if (ch == 'y') {
    goto Met;
  }

  return 0;
}

// Решение системы методом Гаусса

int solution_system(int n, double **A, double *X)
{
  int i, j, k;
  double s, p;

  print_system(n,A,X);

  for (i=0; i<n; i++) {
    s = A[i][i];
    if (s!=0) {
      s = 1.0/s;
      A[i][i] = 1.0;
      for (j=i+1; j<n; j++) A[i][j] *= s;
      X[i] *= s;
      if (i<n-1){
        for (k=i+1; k<n; k++){
          p = A[k][i];
          A[k][i] = 0;
          for (j=i+1; j<n; j++) A[k][j] -= (p*A[i][j]);
          X[k] -= (p*X[i]);
        }
      }
      print_system(n,A,X);
    }
    else {
      return -1;
    }
  }

  for (i=(n-2); i>=0; i--) {
    s = 0;
    for (j=i+1; j<n; j++) s += (A[i][j]*X[j]);
    X[i] -= s;
  }
  print_system(n,A,X);

  return 0;
}

// Вывод расширенной матрицы на экран

void print_system(int n, double **A, double *X)
{
  int i, j;
  for (i=0; i<n; i++){
    for (j=0; j<n; j++) printf("%14.6le ",A[i][j]);
    printf("%14.6le\n",X[i]);
  }
  printf("\n");
  return;
}
```