**Assignment 30-06-2017**

**Storage Classes:**

1. **Auto:**
   1. Default initial value of auto variable is garbage.
   2. Visibility of auto variable is within the block where it has declared.
   3. Scope of auto variable is within the block where it has declared.
   4. An auto variable gets memory at run time.

**Question:**

1. What will be output of the following program and why?

#include<stdio.h>

int main(){

    int a=0;

    {

         int a=10;

         printf("%d",a);

         a++;

         {

             a=20;

         }

         {

             printf(" %d",a);

             int a=30; {a++;}

             printf(" %d",a++);

         }

         printf(" %d",a++);

    }

    printf(" %d",a);

    return 0;

}

1. What will be output of the following program and why?

#include<stdio.h>

int main(){

    int i=0;

    {

         auto int a=20;

         XYZ:;

         printf("%d",a);

         a++;

         i++;

    }

    if (i<3)

         goto xyz;

    return 0;

}

1. What will be output of the following program and why?

#include<stdio.h>

int main(){

    int i=0;

    {

         XYZ:;

         auto int a=20;

         printf("%d",a);

         a++;

         i++;

    }

    if (i<3)

         goto xyz;

    return 0;

}

1. **register:**
   1. All register variable in c stores in CPU not in the memory.
   2. Register can be used only with integer variable.
   3. A register variable execute faster than other variables because it is stored in CPU so during the execution compiler has no extra burden to bring the variable from memory to CPU.
   4. Since a CPU have limited number of register so it is programmer responsibility which variable should declared as register variable i.e. variable which are using many times should declared as a register variable.
   5. We cannot dereference register variable since it has not any memory address.
   6. Default initial value of register variable is garbage.
   7. Scope and visibility of register variable is block.
2. What will be output of following program and why?

int main()

{

  register int i = 10;

  int \*a = &i;

  printf("%d", \*a);

  return 0;

}

1. What will be output of following program and why?

int main()

{

  int i = 10;

  register int \*a = &i;

  printf("%d", \*a);

  return 0;

}

1. What will be output of following program and why?

int main()

{

  int i = 10;

  register static int \*a = &i;

  printf("%d", \*a);

  return 0;

}

1. What will be output of following program and why?

int main()

{

  register float a = 10.5;

  printf("%f", a);

  return 0;

}

1. **Static**
   1. This modifier is used with all data types like int, float, double, array, pointer, structure, function etc.
   2. It is not default storage class of global variables.
   3. Default initial value of static integral type variables are zero otherwise null.
   4. A same static variable can be declared many times but we can initialize at only one time.
   5. We cannot write any assignment statement globally.
   6. A static variable initializes only one time in whole program.
   7. If we declared static variable locally then its visibility will within a block where it has declared.
   8. A static variables or functions have internal linkage. An internal linkage variables or functions are visible to the file where it has declared.

**Question:**

1. #include<stdio.h>int a;

int main(){

    printf("%d",a);

     return 0;

}

1. #include<stdio.h>

static int a;

int main(){

    printf("%d",a);

     return 0;

}

1. #include<stdio.h>

extern int a;

int main(){

    printf("%d",a);

    return 0;

}

1. #include <stdio.h>

static char c;

static int i;

static float f;

static char \*str;

int main(){

    printf("%d %d %f %s",c,i,f,str);

    return 0;

}

1. #include <stdio.h>

static int i;     //Declaring the variable i.

static int i=25;  //Initializing the variable.

static int i;     //Again declaring the variable i.

int main(){

    static int i;    //Again declaring the variable i.

    printf("%d",i);

    return 0;

}

1. #include <stdio.h>

static int i;        //Declaring the variable

static int i=25;     //Initializing the variable

int main(){

         printf("%d",i);

    return 0;

}

static int i=20;     //Again initializing the variable

#include <stdio.h>

static int i=10;   //Initialization statement

i=25;              //Assignment statement

int main(){

    printf("%d",i);

    return 0;

}

**viii)**

#include <stdio.h>

static int i=10;

int main(){

    i=25;       //Assignment statement

    printf("%d",i);

    return 0;

}

**ix)**

#include <stdio.h>

static int i=10;

int main(){

    i=5;

    for(i=0;i<5;i++){

         static int a=10; //This statement will execute

                          //only time.

         printf("%d",a++);//This statement will execute

                          //five times.

    }

    return 0;

}

**x)**

#include<stdio.h>

int main(){

    {

         static int a=5;

         printf("%d",a);

    }

//printf("%d",a);   variable a is not visible here.

    return 0;

}

**xi)**

#include<stdio.h>

static float a=144.0f; //global to all function

int main(){

    {

         printf("%d",a); //variable a is visible here.

       //printf("%d",b); variable b is not visible here.

    }

    printf("%d",a);   //variable a is visible here.

    //printf("%d",b);    variable b is not visible here.

    return 0;

}

static int b=5;    //Global to only calculation function

void calculation(){

    printf("%d",a);   //variable a is visible here.

    printf("%d",b);   //variable b is visible here.

}

**xii)**

//one.c

#include<stdio.h>

static int i=25;

static int j=5;

void main(){

    clrscr();

    sum();

    getch();

}

//two.c

#include<stdio.h>

extern int i; //Declaration of variable i.

extern int j; //Declaration of variable j.

/\*\*

Above two lines will search the initialization statement of variable i and j either in two.c (if initialized variable is static or extern) or one.c (if initialized variable is extern)

\*/

extern void sum(){

    int s;

    s=i+j;

    printf("%d",s);

}

**xiii)**

#include<stdio.h>

void visit();

int main(){

    int i=0;

    {                    //Opening inner block

         static int a=5;  //locally declaration

         XYZ:;            //Label of goto statement

         printf("%d  ",a);

         a++;

         i++;

    }

    visit();

    printf("%d",a);

    if(i<5)

             goto XYZ;

    return 0;

}

void visit(){

}

**xiv)**

//Locally declarations of variable

There are two c source code files:

//one.c

#include<stdio.h>

void main(){

    int i;

    for(i=0;i<3;i++){

         {

             static int a=5;

             printf("%d\n",a);

             a++;

         }

         visit();

    }

    getch();

}

//two.c

#include<stdio.h>

void visit(){

    printf("Don’t disturb, I am learning storage class");

printf("%d",a);

}

1. **extern**
   1. This modifier is used with all data types like int, float, double, array, pointer, structure, function etc.
   2. It is default storage class of all global variables as well all functions.
   3. When we use extern modifier with any variables it is only declaration i.e. memory is not allocated for these variable. Hence in second case compiler is showing error unknown symbol
      1. To define a variable i.e. allocate the memory for extern variables it is necessary to initialize the variables.
   4. If you will not use extern keyword with global variables then compiler will automatically initialize with default value to extern variable.
   5. Default initial value of extern integral type variable is zero otherwise null.
   6. We cannot initialize extern variable locally i.e. within any block either at the time of declaration or separately. We can only initialize extern variable globally.
   7. If we declare any variable as extern variable then it searches that variable either it has been initialized or not. If it has been initialized which may be either extern or static\* then it is ok otherwise compiler will show an error.
   8. A particular extern variable can be declared many times but we can initialize at only one time.
   9. We cannot write any assignment statement globally.
   10. If declared an extern variables or function globally then its visibility will whole the program which may contain one file or many files.

**i)**

#include <stdio.h>

int i;    //By default it is extern variable

int main(){

    printf("%d",i);

    return 0;

}

**ii)**

#include <stdio.h>

extern int i;    //extern variable

int main(){

    printf("%d",i);

    return 0;

}

**iii)**

#include <stdio.h>

void sum(int,int) //By default it is extern.

int main(){

    int a=5,b=10;

    sum(a,b);

    return 0;

}

void sum(int a,int b){

    printf("%d”",a+b);

}

**iv)**

#include <stdio.h>

extern int i=10;    //extern variable

int main(){

    printf("%d",i);

    return 0;

}

#include <stdio.h>

char c;

int i;

float f;

char \*str;

int main(){

    printf("%d %d %f %s",c,i,f,str);

    return 0;

}

#include <stdio.h>

int main(){

extern int i=10; //Try to initialize extern variable

                 //locally.

    printf("%d",i);

    return 0;

}

#include <stdio.h>

int main(){

    extern int i; //Declaration of extern variable i.

    int i=10;   //Try to locally initialization of

                //extern variable i.

    printf("%d",i);

    return 0;

}

#include <stdio.h>

int main(){

    extern int i; //It will search the initialization of

    //variable i.

    printf("%d",i);

    return 0;

}

int i=20;    //Initialization of variable i.

#include <stdio.h>

int main(){

extern int i; //It will search the any initialized

              //variable i which may be static or

              //extern.

printf("%d",i);

    return 0;

}

extern int i=20; //Initialization of extern variable i.

#include <stdio.h>

int main(){

extern int i; //It will search the any initialized

              //variable i which may be static or

              //extern.

printf("%d",i);

return 0;

}

static int i=20; //Initialization of static variable i.

#include <stdio.h>

int main(){

    extern int i;   //variable i has declared but not

                    //initialized

    printf("%d",i);

    return 0;

}

extern int i; //Declaring the variable i.

int i=25;     //Initializing the variable.

extern int i; //Again declaring the variable i.

#include <stdio.h>

int main(){

    extern int i; //Again declaring the variable i.

    printf("%d",i);

    return 0;

}

extern int i; //Declaring the variable

int i=25;     //Initializing the variable

#include <stdio.h>

int main(){

         printf("%d",i);

    return 0;

}

int i=20; //Initializing the variable

#include <stdio.h>

extern int i;

int i=10;   //Initialization statement

i=25;       //Assignment statement

int main(){

    printf("%d",i);

    return 0;

}

#include <stdio.h>

extern int i;

int main(){

    i=25;       //Assignment statement

    printf("%d",i);

    return 0;

}

int i=10;   //Initialization statement

//one.c

#include<stdio.h>

int i=25; //By default extern variable

int j=5;  //By default extern variable

/\*\*

Above two line is initialization of variable i and j.

\*/

void main(){

    clrscr();

    sum();

    getch();

}

//two.c

#include<stdio.h>

extern int i; //Declaration of variable i.

extern int j; //Declaration of variable j.

/\*\*

Above two lines will search the initialization statement of variable i and j either in two.c (if initialized variable is static or extern) or one.c (if initialized variable is extern)

\*/

void sum(){

    int s;

    s=i+j;

    printf("%d",s);

}