CSE 1320

Exam 2 Review

Instructor : Donna French

Arrays of Pointers

```
char *Question[] = {"What","are","you","doing","?"};
int i=4;

do
{
    printf("%s ", *(Question+i--));
}
while (i > 0);
```

```
? doing you are
```

fgets()

Given

```
char InputBuffer[326];
fgets(InputBuffer, 500, stdin);
```

325

What is the max length of a string that could be stored in InputBuffer?

What value of `fp` is used to indicate the keyboard?

```
fgets(inbuff, n, fp);
```

stdin

Given this code snippet

```
char InputBuffer[20];
char *InputPtr;
InputPtr = fgets(InputBuffer, 10, stdin);
printf("%s", InputPtr);
```

What would print if the following was given as input when prompted.

```
supercalifragilisticexpialidocious
```

supercali

return vs exit

```
void MyFunction(void)
{
    printf("Hello");
    return;
}

MyFunction();
printf("Goodbye");
```

```
HelloGoodbye
```

```
void MyFunction(void)
{
    printf("Hello");
    exit(0);
}

MyFunction();
printf("Goodbye");
```

```
Hello
```

do while continue break

```
int i = 10;
do
{
     printf("%d\n", i);
     if (i % 3)
     {
          i-=1;
          continue;
     }

     break;
}
while (i > 0);
printf("i = %d", i);
```

```
10
9
i = 9
```

include guard

```
#ifndef COKE_LIB_H
#define COKE_LIB_H

void MyFunction(int, int, char, long);

#endif
```

Structures

```
struct Dog
{
    char name[20];
    char breed[20];
    int age;
    float weight;
};

struct Dog Fluffy;

strcpy(Fluffy.name, "Fluffy");
strcpy(Fluffy.breed, "structure");
Fluffy.age = 0;
Fluffy.weight = 1000;

printf("Don't struct %s", Fluffy.name);
```

Arrays of Structures

```
void FillInName(struct Dog *LitterElement, char *FluffyName)
{
   strcpy(LitterElement->name, FluffyName);
}

void FillInBreed(struct Dog Litter[])
{
   strcpy(Litter[0].breed, "structure");
   strcpy(Litter[1].breed, "structure");
}
```

```
struct Dog
{
   char name[20];
   char breed[20];
   int age;
};
```

```
----------------------------------------
FillInName(          , "FluffyJr");
FillInName(          , "LittleMissFluffy");
FillInBreed(      );
```

```
printf("%s is a %s that is %d days old\n",
       Litter[0].name, Litter[0].breed,
       Litter[0].age);
printf("%s is a %s that is %d days old\n",
                        ame, Litter[1].breed,
                        ge);
```

```
FluffyJr is a structure that is 0 days old
LittleMissFluffy is a structure that is 0 days old
```

Unions

Given this code snippet, what would print?

```
union Pizza
{
    int pepperoni;
    int tomato_sauce;
    long crust;
    double cheese;
};

union Pizza Large;
```

```
Large.crust = 200;

Large.pepperoni = 10;

Large.cheese = 34;

Large.tomato_sauce = 123;

printf("%d", Large.cheese);
```

123

typedefs

```
enum Colors {yellow, white, red, orange};

typedef int Candy;
Candy Corn = yellow+white+orange;
Candy Apples = red;
Candy Cane = red + white;
printf("%d", Corn+Apples+Cane);
```

9

makefile

```
SRC1 = Code3_1000074079.c
SRC2 = MyLib.c
OBJ1 = $(SRC1:.c=.o)
OBJ2 = $(SRC2:.c=.o)
EXE = $(SRC1:.c=.e)

HFILES = MyLib.h

CFLAGS = -g
```

```
all : $(EXE)

$(EXE): $(OBJ1) $(OBJ2)
	gcc $(CFLAGS) $(OBJ1) $(OBJ2) -o $(EXE)

$(OBJ1) : $(SRC1) $(HFILES)
	gcc -c $(CFLAGS) $(SRC1) -o $(OBJ1)

$(OBJ2) : $(SRC2) $(HFILES)
	gcc -c $(CFLAGS) $(SRC2) -o $(OBJ2)
```

```
char ch1, ch2;
int sum = 0;

printf("Enter a character ");
scanf(" %c", &ch1);
printf("Enter a character ");
scanf(" %c", &ch2);

sum = ispunct(ch1) ? sum+1 : sum-1;
sum = isdigit(ch2) ? sum-2 : sum+2;
sum = isalpha(ch1) ? sum*3 : sum/3;
sum = isupper(ch2) ? sum/4 : sum*4;
sum = islower(ch2) ? sum+5 : sum-6;
sum = isalnum(ch1) ? sum*7 : sum/8;

printf("%d", sum);
```

```
1
&

sum
---
 0
 0 - 1 = -1
-1 + 2 = 1
 1 / 3 =  0
 0 * 4 = 0
 0 - 6 = -6
-6 * 7
```

-42

```
char ch;
int sum = 0;

printf("Enter a character ");
scanf(" %c", &ch);

sum = islower(ch) ? sum+1 : sum-1;
sum = isupper(ch) ? sum-2 : sum+2;
sum = isalpha(ch) ? sum*3 : sum/3;
sum = isalnum(ch) ? sum/4 : sum*4;
sum = isdigit(ch) ? sum+5 : sum-6;
sum = ispunct(ch) ? sum*7 : sum/8;

printf("%d", sum);
```

A

```
sum
---
 0
 0 - 1 = -1
-1 - 2 = -3
-3 * 3 = -9
-9 / 4 = -2
-2 - 6 = -8
-8 / 8 = -1
```

-1

Double Indirection

```
enum Kingdom {King, Queen, Prince, Princess=18};

int Heir[2] = {Prince, Princess};
int *FutureMonarch = &Heir[Queen];
int **RulerOfAll = &FutureMonarch;
printf("%d", **RulerOfAll);
```

18

enumeration

```
enum {DontStudy, StudySome, Study=4, StudyALot} StudyStatus;
char grade;
int Choice = 5;

switch (Choice)
{
   case DontStudy : grade = 'D'; break;
   case StudySome : grade = 'C'; break;
   case Study     : grade = 'B'; break;
   case StudyALot : grade = 'A'; break;
   default : grade = 'F';
}
printf("Grade = %c", grade);
```

```
Grade = A
```

`for` loop

Fill out the for loop needed to print the following output

```
i = 8
i = 4
i = 2
```

```
int i;

for (i = 8; i > 1; i /= 2)
   printf("i = %d\n", i);
```

`for` loop

```
int Sonic = 1, Tails = 1, Amy = 22;
float Knuckles = 0.2;
for (Sonic = 12; Tails < 9; Sonic=Sonic/3, Tails*=2, Amy=Amy>>1)
Knuckles += 0.2;
printf("%d%d%d%1.1f", Sonic, Tails, Amy, Knuckles);
```

```
Sonic Tails Amy   Knuckles
12    1     22    0.2 + 0.2 = 0.4
4     2     11    0.2 + 0.4 = 0.6
1     4     5     0.2 + 0.6 = 0.8
0     8     2     0.2 + 0.8 = 1.0
0     16    1
```

01611.0

String Library

Given this code snippet, what will print?

```
char Array[100] = "texas";
char Ch1   = 'e';
char Ch2[4] = "xas";

*(strstr(Array, Ch2)) = *(strchr(Array, Ch1));

*(strchr(Array, Ch1)) = *Array;

printf("%s", Array);
```

tteas

String Library

Given this code snippet, what will print?

```
char Array1[100] = "they all";
char Array2[100] = "y";
char Array3[100] = " ";
char *FirstOccur;


FirstOccur = strpbrk(Array1, Array3);
while (FirstOccur != NULL)
{
   *FirstOccur = 'm';
   FirstOccur = strpbrk(Array1, Array3);
}
FirstOccur = strpbrk(Array1, Array2);
while (FirstOccur != NULL)
{
   *FirstOccur = ' ';
   FirstOccur = strpbrk(Array1, Array2);
}

printf("%s\n", Array1);
```

the mall

String Library

```
char Array1[100] = "Is6this7test8hard?";
char Array2[100] = "678";
char *Token;

Token = strtok(Array1, Array2);

while (Token != NULL)
{
   printf("%s\n", Token);
   Token = strtok(NULL, Array2);
}
```

```
Is
this
test
hard?
```

String Library

Given this code snippet, fill in the blanks to make "Array1 and Array2 match" print.

```
char Array1[100] = "hello";
char Array2[100] = "hellz";
if (strncmp(Array1, Array2, 4) == 0)
    printf("Array1 and Array2 match");
```

toupper and tolower

What would this code snippet print?

HeLLO

```
char Word[10] = "OlshjfeLd";


tolower(Word[0]);


printf("%c%c%c%c%c", toupper(Word[3]),
tolower(Word[6]), toupper(Word[1]), toupper(Word[7]),
Word[0]);
```

True or False?

Dereferencing a NULL pointer may result in a segmentation fault.

FALSE – It will result in a segmentation fault every time.

An array in C must be null terminated

FALSE – An array must be null terminated to be treated like a string by the string library functions.

When an array is passed to a function, a copy of the array is passed and used within the function.

FALSE – the address is passed.

When the compiler sees a sequence of characters enclosed in double quotes, it stores the sequence and appends a terminating ‘\n’ to the end of the character sequence.

FALSE – appends a terminating `'\0'`

atof() and atoi()

Given this code snippet, what would print?

```
char MyLetter[6] = "-01.2";
float MyNumber = 0.0;
int MyOtherNumber = 0;

MyOtherNumber = atoi(MyLetter);
MyNumber = atof(MyLetter);
printf("%d%2.2f%.2f", MyOtherNumber, MyNumber, atoi(MyLetter) * atof(MyLetter));
```

-1

-1.2

-1 -1.2 -1 -1.2

-1-1.201.20

Storage Class and Scope

Changing the storage class from `auto` to `static` also changes the scope of the variable.

FALSE – changing storage class does not change the scope of the variable but it does change the lifetime of the variable.

The scope of a `static` variable can be local or global depending on where it is declared.

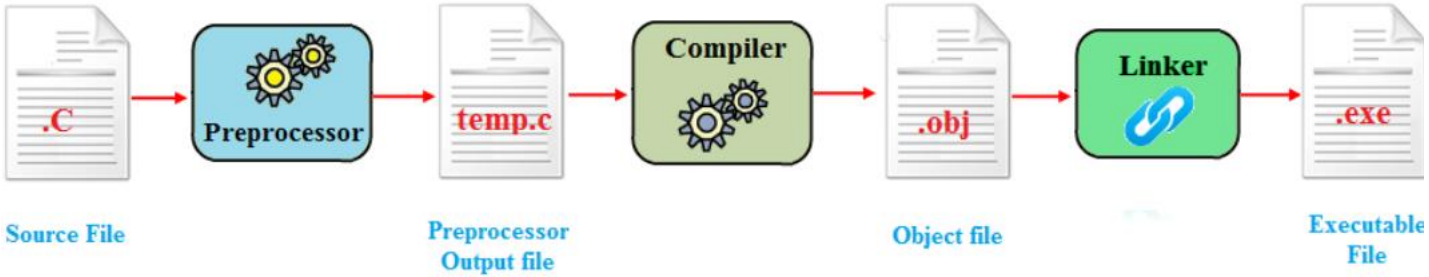
TRUE – using storage class `static` does not change the scope of the variable.

The scope of an `auto` variable can be local or global depending on where it is declared.

TRUE – the scope of any storage class is determined by where it is declared.

Memory space for a `static` variable is allocated when the function is called where the variable is declared and is deallocated when program ends.

TRUE – `auto` variables deallocate when the function ends but `static` do not



creates an object file

compiler

takes in object files and produces an executable file

linker

```
SRC1 = Code4_1000074079.c
SRC2 = MyLib.c
OBJ1 = $(SRC1:.c=.o)
OBJ2 = $(SRC2:.c=.o)
EXE = $(SRC1:.c=.e)

HFILES = MyLib.h

CFLAGS = -g

all : $(EXE)

$(EXE): $(OBJ1) $(OBJ2)
        gcc $(CFLAGS) $(OBJ1) $(OBJ2) -o $(EXE)

$(OBJ1) : $(SRC1) $(HFILES)
        gcc -c $(CFLAGS) $(SRC1) -o $(OBJ1)

$(OBJ2): $(SRC2) $(HFILES)
        gcc -c $(CFLAGS) $(SRC2) -o $(OBJ2)
```

Exam 2 Review

storage class used to associate an identifier with a type

`typedef`

statement used in `main()` to terminate the program

`return`

`register, auto, static`

storage class

Exam 2 Review

default storage class for multidimensional arrays

`auto`

statement used to terminate a program from anywhere in the program

`exit()`

can concurrently hold multiple data values

structure

Exam 2 Review

T / F When the compiler sees a sequence of characters enclosed in double quotes, it stores the sequence and appends a terminating `'\n'` to the end of the character sequence.

What would print given the following input?

university

```
char InputBuffer[20];
fgets(InputBuffer, 10, stdin);
printf("%s", InputBuffer);
```

```
universit
```

Exam 2

F A `union` is the same as a structure in that it will contain the distinct value of each of its members at any given time.

T Changing the storage class from `auto` to `static` will not change the scope of the variable.

T Automatic variables are created each time the function in which they are declared is called and are destroyed when that function terminates and, when an automatic variable is created with an initialization, the initialization is done every time the function is called.

T Static variables exist the whole time the program is executing and memory space for a static variable is allocated when the function in which it is declared is called and is deallocated when program ends.

F An array in C must be null terminated.

Exam 2

```
char Pie[10] = "+3.14";
printf("%3.3f%d", atof(Pie), atoi(Pie));
```

What would print? `3.1403`

What are the first two commands in debug?

`break main` `run`

This statement, Minnie->shoe_color can also be written as

`(*Minnie).shoe_color`

Exam 2

Fill in the blanks to complete the include guard

```
#_____ifndef_____ COKE_LIB_H

#_____define_____ COKE_LIB_H

void MyFunction(int, int, char, long);

#_____endif_____
```

Exam 2

3. Choose what would print

```
char ch1 = 'A', ch2 = 'b', ch3 = '9', ch4 = '*';
int sum = 0;

sum = ispunct(ch4) ? sum+1 : sum-4;
sum = isalpha(ch3) ? sum*2 : sum/5;
sum = islower(ch2) ? sum+3 : sum-6;
sum = isdigit(ch4) ? sum-4 : sum+7;
sum = isalnum(ch3) ? sum*5 : sum/8;
sum = isupper(ch1) ? sum/6 : sum*9;
printf("%d", sum);
```

a. 8

b. 10

c. 1

d. 0

Exam 2

4. Choose what would print

```
char *Halloween[5] = {"Candy", "Costumes", "Pumpkins", "Scary", "  "};
printf("%c%c%c%c%c%c%c%c%c%c%c%c%c%c",
        toupper(Halloween[1][3]), *(*(Halloween + 3) + 3), Halloween[2][5],
        tolower(*Halloween[0]), Halloween[2][4], tolower(Halloween[4][1]),
        Halloween[1][1], Halloween[3][3], toupper(*(*(Halloween + 4))),
        toupper(*(*(Halloween + 1) + 3)),Halloween[3][3],Halloween[1][6],
        *(*(Halloween + 0) + 1), *(*(Halloween + 1) + 3));
```

a. Halloween b. Trick or Treat c. Trick Or Treat d. NmuctSCmDNm Cn

Exam 2

5. Choose what would print

```
int TootsiePop[10] = {876,-95,6,102,3,-4,-5,10,4,-9};
int i, count = 3;
for (i = 1; i < *&*&*&*&*&*&*&*&*&*(TootsiePop+4); i++)
count -= *&*&*&*&*&*&*&*&*&*&*&*&*&*(TootsiePop+i);
printf("How many licks does it take to get the center of a Tootsie Pop?\n");
printf("%d", count/30);
```

a. 3.06	b. 92	c. 3	d. Would not compile

Exam 2

6. Choose what would print

```
enum Government {Local=100, State, Federal};

int Arlington[3] = {Local, State, Federal};
int *Texas = Arlington+1;
int **USA = &Texas;
printf("%d", **USA);
```

a. 1

b. 101

c. 100

d. Would not compile

Exam 2

7. Choose what would print

```
int Shuri = 1, Okoye = 3, Ramonda = 22;
float TChalla = -15.5;
for (Shuri = 16; Okoye < 9; Shuri/=4, Okoye*=2, Ramonda=Ramonda>>1)
{
    TChalla += 0.5;
}
printf("%d%d%d%1.1f", Shuri, Okoye, Ramonda, TChalla);
```

a. 1125-14.5 b. 4611-14.5 c. 1125-14.0 d. 1125-15.0

Exam 2

8. What would print?

```
enum function {fly, fall, swim, hold, tear=1, sink, walk, crawl, spill};

typedef int Paper;
Paper Plane = fly + fall;
Paper Bag = hold + tear + spill;
Paper Boat = swim + sink;
printf("%d %d %d", Plane, Bag, Boat );
```

a. Would not compile

b. 1 9 4

c. 3 10 5

d. 194

Exam 2

9. What would print?

```
char Array1[100] = "How now brown cow";
char Array2[100] = "ow";
char Array3[100] = {};
char Array4[100] = " ";
char *Token;
char *FirstOccur;

Token = strtok(Array1, Array2);

while (Token != NULL)
{
    strcat(Array3, Token);

    Token = strtok(NULL, Array2);
}

FirstOccur = strpbrk(Array3, Array4);
while (FirstOccur != NULL)
{
    *FirstOccur = '_';
    FirstOccur = strpbrk(Array3, Array4);
}
```

a. Would not compile

b. H n brn c

c. H_n_brn_c

d. _ow__ow___ow___ow

Exam 2

10. What would print?

```
char Array[100] = "shape";
char Ch1   = 'h';
char Ch2[4] = "ape";

*(strstr(Array, Ch2)) = *(strchr(Array, Ch1) + 3);
*(strchr(Array, Ch1) + 3) = *(strchr(Array, Ch1) + 2);
*(strchr(Array, Ch1) + 2) = *(strchr(Array, Ch1) + 1);

printf("%s", Array);
```

a. shapa

b. shepe

c. sheep

d. shape

Two's Complement

Write the binary value of two's complement for

0111	flip the bits - 1000	add 1	1000 + 1	1001
0011	flip the bits - 1100	add 1	1100 + 1	1101
0001	flip the bits - 1110	add 1	1110 + 1	1111