11.04.19

POINTERS AND REFERENCES WORKSHEET

1) Write a function that uses pointers to copy an array of double.
Double türündeki bir diziyi kopyalamak için işaretçileri kullanan bir fonksiyon yazın.

2) Write a function that uses pointers to search for the address of a given integer in a given array. If the given integer is found, the function returns its address; otherwise it returns NULL.
Belirli bir dizideki belirli bir tamsayı adresini aramak için işaretçileri kullanan bir fonksiyon yazın. Verilen tam sayı bulunursa, fonksiyon tamsayının adresini döndürür; aksi halde NULL döndürür.

3) Write a function that is passed an array of n pointers to floats and returns a newly created array that contains those n float values.
Parametre olarak n tane float pointer içeren bir dizi ve n değerini alan ve verilen bu n tane float değeri yeni bir float dizisine kaydedip döndüren bir fonksiyon yazın.

4) Write a function that returns the *numerical derivative* of a given function *f* at a given point *x*, using a given tolerance *h*. Use the formula

$$f'(x) = (f(x + h) – f(x – h)) / (2*h)$$

Your function prototype is

```
double derivative(double (*)(double), double, double)
```

Belirli bir fonksiyonun (f) sayısal türevini, verilen bir toleransı (h) kullanarak verilen bir noktada (x) hesaplayan bir fonksiyon yazın.

5) Write a function that is passed an array of n pointers to floats and returns a pointer to the maximum of the n floats.
Parametre olarak n tane float pointer içeren bir dizi ve n değerini alan ve dönüş değeri olarak bu n tane float değerden maximum olana pointer döndüren bir fonksiyon yazın.

```
float* max(float* p[], int n)
void print(float [ ], int);

void print(float* [ ], int);
```

6) Write the following function that is passed an array of n pointers to floats and returns a newly created array that contains those n float values in reverse order.
Prototipi aşağıda verilen ve parametre olarak gönderilen float pointer dizisinin tersini yeni bir dizi olarak döndüren fonksiyonu yazın.

```
float* mirror(float* p[],int n)
```

7) Write the following function that returns the number of bytes that s has to be incremented before it points to the null character '\0':
Prototipi aşağıda verilen ve '\ 0' boş karakterine işaret etmeden önce artırılması gereken bayt sayısını döndüren aşağıdaki işlevi yazın:

```
unsigned len(const char* s)
```

8) Write the following function that copies the first *n* bytes beginning with *s2 into the bytes beginning with *s1, where *n* is the number of bytes that s2 has to be incremented before it points to the null character '\0':

```
void cpy(char* s1,con st char* s2)
```

9) Write the following function that copies the first *n* bytes beginning with *s2 into the bytes beginning at the location of the first occurrence of the null character'\0' after *s1, where *n* is the number of bytes that s2 has to be incremented before it points to the null character

'\0': *** S2 ile başlayan ilk n baytı * s1 ile başlayan baytlara kopyalayan aşağıda prototipi verilen fonksiyonu yazın; burada n, s2'nin boş karaktere işaret etmeden önce artırılması gereken bayt sayısıdır.**

void cat(char* s1,con st char* s2)

10) Write the following function that compares at most *n* bytes beginning with s2 with the corresponding bytes beginning with s1, where *n* is the number of bytes that s2 has to be incremented before it points to the null character '\0'. If all *n* bytes match, the function should return 0; otherwise, it should return either -1 or 1 according to whether the byte from s1 is less than or greater than the byte from s2 at the first mismatch.

int cmp(char* s1,char * s2)

S2 ile başlayan en çok n baytı, s1 ile başlayan ilgili baytlarla karşılaştıran prototipi verilen fonksiyonu yazın; burada n, s2'nin '\ 0' boş karakterine işaret etmeden önce artırılması gereken bayt sayısıdır. Tüm n bayt eşleşirse, işlev 0 döndürmelidir; Aksi takdirde, ilk uyuşmazlıktaki s1'den gelen baytın, s2'den gelen bayttan küçük veya küçük olmasına bağlı olarak -1 veya 1 döndürmelidir.

11) Write the following function that returns the sum of the floats pointed to by the first n pointers in the array p:

float sum(float* p[],int n)

Prototipi verilen, parametre olarak aldığı float pointer dizisinin ilk n elemanının toplamını döndüren fonksiyonu yazın.

12) Write the following function that changes the sign of each of the negative floats pointed to by the first n pointers in the array p:

void abs(float* p[],int n)

Prototipi verilen, parametre olarak aldığı float pointer dizisinin ilk n elemanında bulunan negatif sayıların işaretini değiştiren fonksiyonu yazın.

13) Write the following function that indirectly sorts the floats pointed to by the first n pointers in the array p by rearranging the pointers:

void sort(float* p[],int n)

Pointerları yeniden düzenleyerek p dizisindeki ilk n pointerın gösterdiği değişkenleri dolaylı olarak sıralayan prototipi verilen fonksiyonu yazın.

14) Implement the Indirect (using pointers) *Selection Sort* using an array of pointers.
Seçmeli Sıralama algoritmasını pointer array kullanarak yazın.

15) Implement the Indirect (using pointers) *Insertion Sort*.
Eklemeli Sıralama algoritmasını pointer array kullanarak yazın.

16) Apply the derivative() function to the following functions defined in <math.h>:
Aşağıda belirtilen <math.h> kütüphanesinde tanımlı fonksiyonları derivative() fonksiyonuna uygulayınız.
a. sqrt(),at the point x = 4;
b. cos(),at the point x = p/6;
c. exp(),at the point x = 0;
d. log(),at the point x = 1.

17) Write the following function that returns the product of the n values *f*(1), *f*(2), ..., and *f*(*n*).
Prototipi aşağıda belirtilen, n tane değerin (*f*(1), *f*(2), ..., ve *f*(*n*).) çarpımını döndüren fonksiyonu yazınız.

int product(int (*pf)(int k),int n)