**Karadeniz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü**
**2017-2018 Bahar Dönemi, Programlamaya Giriş Bütünleme Sınavı**
**19/06/2018, Saat: 10:00, Süre: 2 saat**

**AD-SOYAD:** **ÖĞR. NO:** **İMZA:**

1. Aşağıda verilen `main()` fonksiyonunun çıktısını bulunuz. `func()` fonksiyonunun ne yaptığını ***bir cümleyle*** ifade ediniz. (**30p**)

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

void func(char * input){

   printf("%c", *input);

   if(*(input+1)  != '\0')
    func(input+1);
   else
     printf("-");

   printf("%c", *input);
}

int main()
{
    char  *in = "Hello World";
    func(in);

    return 0;
}
```

Hello World-dlroW olleH

2. Aşağıda verilen `stringCompare()` fonksiyonundan, s1 parametresinin işaret ettiği dizi ile s2 parametresinin işaret ettiği dizinin maksimum n karakterini birbiriyle kıyaslaması ve kıyaslama sonucunda, s1 dizisinin s2 dizisinden alfabetik olarak daha küçük olması, daha büyük olması ve eşdeğer olmaları durumlarında sırasıyla -1, 1 ve 0 değerlerini geriye döndürmesi beklenmektedir (Örnek: `stringCompare("abcde", "abcdf", 4)`, `stringCompare("abcde", "abcff", 4)` ve `stringCompare("abcfe", "abcdf", 4)` çağrıları sırasıyla 0, -1 ve 1 değerlerini geriye döndürmelidir). İşaret edilen dizilerin en az n adet karaktere sahip olduğunu varsayınız.

   Fonksiyonun 3 adet mantıksal hata içerdiği için doğru çalışmadığını göz önünde bulundurarak, hatalı kod bölümünü işaretleyerek düzeltilmiş halini yazınız. (**Not:** *i)* ASCII tablodaki karakter kodlarının alfabetik sıraya göre küçükten büyüğe doğru sıralandığını göz önünde bulundurunuz, *ii)* Düzeltme işlemi, yalnızca mevcut bir kod kısmının yerine doğrusunun yazılması ya da tamamıyla silinmesi ile gerçekleştirilir, *iii)* Düzeltilmeyen hatalara hiç bir puan verilmeyecektir, i*v)* Kodun sil-baştan yazılmasına ya da mevcut koda yeni satırların eklenmesine izin yoktur, *v)* Herhangi bir hata içermeyen kodun hatalı kod ile birlikte işaretlenmesi durumunda cevap geçersiz sayılacaktır) (**35p**)

```
int stringCompare( const char *s1, const char *s2, int n )
{
     int counter;

     for ( counter = 0; counter < n && (*s1 == *s2); counter++, s1++, s2++ ){
          ;
     }

    s1--;
    s2++;

     if ( *s1 == *s2 )
          return 0;
     else if ( *s1 < *s2 )
          return -1;
     else
          return 1;
}
```

3. `decideRightTriangle()` fonksiyonu, kendisine adresleri verilen x-y koordinat eksenindeki üç noktanın bir dik üçgenin köşelerine karşılık gelmesi durumunda geriye 1, aksi taktirde 0 değerini döndürmektedir. `calculateSquaredDistance()` fonksiyonu ise kendisine adresleri verilen iki nokta arasındaki uzaklığın karesini geriye döndürmektedir. Buna göre koddaki boşlukları uygun şekilde doldurunuz. (**Not:** *i)* Her bir boşluğa yalnızca tek bir kod satırı yazılmalıdır, *ii)* Üst alma işlemi için `pow()` fonksiyonu kullanılabilir) (**35p**)

```
struct point {
    int x;
    int y;
};

int decideRightTriangle(struct point *point1, struct point *point2, struct point *point3){
    double squaredDistances[3];
    int counter;

    squaredDistances[0] = calculateSquaredDistance(point1, point2);
    squaredDistances[1] = calculateSquaredDistance(point2, point3);
    squaredDistances[2] = calculateSquaredDistance(point3, point1);

    int maxIndex = 0;

    for(counter = 1; counter < 3; counter++){
        if(squaredDistances[counter] > squaredDistances[maxIndex]){
            maxIndex = counter;
        }
    }

    double sum = 0;

    for(counter = 0; counter < 3; counter++){
        if(counter != maxIndex){
            sum += squaredDistances[counter];
        }
    }

    if(squaredDistances[maxIndex] == sum){
        return 1;
    }else{
        return 0;
    }
}

double calculateSquaredDistance(struct point *point1, struct point *point2){
    return pow((point1->x) - (point2->x), 2) + pow((point1->y) - (point2->y), 2);
}
```