Assume that the sizes of the data types (in bytes) are as follows: **`char`** `= 1,` **`short`** `= 2,` **`int`** `= 4,` **`long`** `= 8,` **`float`** `= 4,` **`double`** `= 8.` You may also assume that pointers are 8 bytes and all necessary header files have been included in the code snippets.

1. What is `sizeof("DP")`? Careful with this one!

2. Given the valid declarations below, what is printed?

```
int a[] = {15, 13, 2, 7, 9, 1, 8, 3, 6};
int *p = &a[3];
int *q = &a[5];
```

```
a) printf("%i", q - p);
b) printf("%i", *(q - 3));
c) printf("%i", *q - *p);
d) printf("%i", *(p + 3));
```

3. Given the legal code below, what is printed?

```
char str[25];

strcpy(str, "Four score and seven");
strcpy(&str[6], "crab");
strcat(str, "ble squares.");
puts(str);
```

4. What does the following code snippet print? (Hint: Use a precedence chart. Draw a diagram to help you.)

```
char s[] = "Sghmj";
char *p;

for (p = &s[4]; p >= s; p--)
  ++*p;
puts(s);
```

5. Given the declarations below, give the precise type of each expression **AND** the value of the expression. If the expression is illegal, write **ILLEGAL** in the type column and leave the value blank. Assume that the address of the array **`a`** is 100 and the address of **`p`** is 200. Note that p doesn't change as there are no assignments or side-effect operations. (Hint: Draw a diagram to help you with the questions. The first two expressions are examples.)

```
int a[10] = {2, 1, 9, 0, 4, 7, 6, 5, 8, 3};
int *p = a + 3;
```

| | **Expression** | **Type of the expression** | **Value** |
|---|---|---|---|
| | `p` | **`pointer to int`** | **112** |
| | `p[2]` | **`int`** | **7** |
| a) | `p + 3` | | |
| b) | `*p` | | |
| c) | `&p` | | |
| d) | `p[-1]` | | |
| e) | `*p + 5` | | |
| f) | `*p[0]` | | |
| g) | `&p[4]` | | |
| h) | `*(p + 6)` | | |