HW06

Name: Abdulelah Al shalhoub ID: 435105334

Q1.

1)

```
double *temperatures = (double*) malloc(365*sizeof(double));
```

//

2)

```
char *q = (char*) malloc(14*sizeof(char));
if( q != NULL)
   strcpy(q, "So many books");
```

//

3)

```
int **grades = (int**) malloc(2*sizeof(int*));
if( grades != NULL ){
   grades[0] = (int*) malloc(3*sizeof(int));
   grades[1] = (int*) malloc(3*sizeof(int));
}
if(grades[0] != NULL && grades[1] != NULL){
   grades[0][0] = 36;
   grades[0][1] = 24;
   grades[0][2] = 26;
   grades[1][0] = 81;
   grades[1][1] = 30;
   grades[1][2] = 74;
}
```

//

4)

```
float *zeros = (float*)calloc(4,sizeof(float));
```

//

5)

```
char **names = (char**) malloc(2*sizeof(char*));
int i;
if( names != NULL ){
   names[0] = (char*) malloc(4*sizeof(char));
   names[1] = (char*) malloc(5*sizeof(char));
}
if(names[0] != NULL && names[1] != NULL){
   strcpy(names[0], "Ali");
   strcpy(names[1], "Omar");
}
```

Q2.

1)

```
char *p = "Hello world";
printf("%d\n", *p);
free(p);
```

the code will run fine but,
it will cause **undefined behavior,** it's wrong to free a static pointer
we can correct it by either remove the free or change p to dynamic pointer

```
char *p = (char*) malloc(12*sizeof(char));
 if( p != NULL)
      strcpy(q, "Hello world");
printf("%d\n", *p);
free(p);
```

2)

```
int* p = (int*)malloc(10*sizeof(int));
int* q = (int*)realloc(p, 5*sizeof(int));
free(q);
free(p);
```

the code will run fine but,
realloc already freed p so by calling free(p) so that mean we called free twice and that will result in double free memory corruption.
Correct: just remove free(p)

```
int* p = (int*)malloc(10*sizeof(int));
int* q = (int*)realloc(p, 5*sizeof(int));
free(q);
```

3)

```
int** p = (int*)malloc(2*sizeof(int));
p[0] = (int*)malloc(5*sizeof(int));
p[1] = (int*)malloc(5*sizeof(int));
free(p);
```

the code will run fine but,
doing this will just free p and will not free p[0] and p[1]
correct:

```
int** p = (int*)malloc(2*sizeof(int));
p[0] = (int*)malloc(5*sizeof(int));
p[1] = (int*)malloc(5*sizeof(int));
free(p[0]);
free(p[1]);
free(p);
```