```
{
        int a[3] = { 25,26,27 };
        int* const p = a;
        p++;
        std::cout << *p << std::endl;
}
```

a. 25
b. 26
c. 27
[✓ d. compilation error]

Q6. What is the output of the following program:

```
void main()
{
        int a[3] = { 25,26,27 };
        const int* p = a;
        p++;
        std::cout << *p << std::endl;
}
```

1

a. 25
[✓ b. 26]
c. 27
d. compilation error

Q7. If you have in your program a function Twice that matches the signature of the function pointer declared in Q4, then how would you let the function pointer of Q4 point to Twice function?

1

fb = &Twice;

Q8. Imagine that you have Student structure. Write the code as per the comments below to demonstrate proper allocation and disposal of free store memory. [5]

```
// create an array of 10 Student pointers on free store / heap
```

int** arr{new int*[10]};

```
// write for loop to create 10 Student nodes on free store and keep them
in the array created above
```

for (int i{0}; i < 10 ; i++) {
~~*(*arr+i) = new Student{};~~
arr[i] = new Student{};
}

```
}        =    new *int [10];
```

```
/* imagine that the Student objects are getting used by code represented
by this comment */

// we are done with using the Student objects. Dispose all Student objects
  for(int i{0}; i < 10 ; i++) {
          delete( * arr[i]);
          arr[i] = nullptr;
  }
  delete (arr);
  arr = nullptr;
```

5

Q9. We are building an IoT app which reads data from two sensors – `Temperature` and `Wind`. Both of these sensors send data to app in data type named Input. Complete the following declarations by filling in appropriate keyword (`struct` or `union`). So that these data types could be used in the IoT app to get and process data from either of the sensors. [2]

```
enum class WeatherDataType
{
  Temperature, Wind
};
struct TempData
{
  int stationId;
  time_t time;
  double current;
  double max;
  double min;
};
struct WindData
{
  int StationId;
  time_t time;
  int speed;
  short direction;
};
union Input
{
  WeatherDataType type;
  struct type1;
  {
    TempData temp;
    WindData wind;
  };
};
```

1

void main()
{

...of the following program:

ent objects ar

the Student o

Q10. Implement the FindLongestWord function declared below. It takes a const pointer to a null terminated array of characters and returns the length of largest word in that string. Note that the argument type is a const char*, so you cannot do like the following:

`*s = 'x'; // compilation error, cannot modify the characters of string pointed to by s.`

`char *p {s}; // compilation error, cannot have non-const pointer initialized by const pointer.` **[6]**

6

```
size_t FindLongestWord(const char* s){
    size-t maxLen{0};
    char* start {static-cast <char*> s};
    char* end {nullptr};                   // Pointers for start and
                                           // end of words.
   while(*start != '\0' [illegible]){
     end = start;

     while(*end != '\0' && *end != ' '){  // space character.
        end++;
     }
   ~~int~~ size-t lenWord{end -start};     // ~~Rep~~ Overwrite maxLen
   if(lenWord > maxLen){                   // if word is longest
        maxLen = lenWord;                  // so far
   }

   if(*end == '\0') break;                 // If last word break.

   // Move start pointer to start of next word.
   start = end + 1;
   while(*start == ' '){                  // Skip multiple space characters.
      start++;
   }
  }
  return maxLen;
}
```

ads data from two
data type named
ord (struct or
data from either

Instructions:

1. There are 10 questions in this paper. Q1 to Q7 are 1 mark each. Marks for remaining questions are mentioned against them.
2. For multiple-choice questions you must circle your answer, like, ©.
3. The answers must be written in the question paper itself in the space provided below every question.
4. Work on your solution in the supplementary sheet provided and write your final program as your answer in this paper.
5. To make it simpler, some questions have hardcoded data to work on. But your program logic should not assume hardcoded data and should work on data input in similar format.
6. Using modern C++ features carries marks.
7. You can assume appropriate headers included ie you do not need to write `#include` statements

0

Q1. Which of the following is correct way to declare a 3-d array threeD of integers with 3 rows and 4 columns and 5 pages?

a. `int threeD [3][4][5]`
[✓ b.] `int threeD [5][3][4]`
c. `int threeD [4][3][5]`
d. `int threeD [3][5][4]`

1

Q2. Which of the following is correct way to declare a 'page' pointer p to array in Q1?

a. `int *p[3][4]`
b. `int *p[5][4]`
c. `int (*p)[5][4]`
[✓ d.] `int (*p)[3][4]`

Q3. What will you write in the following blank, if you were to declare a 'row' pointer named rowP and make it point to the 2nd row of 3rd page of array in Q1 using the 'page' pointer in Q2?

int (*p) rowP[4] ~~{*p[2]};~~ {*(p+2)};
rowP++;

Q4. Which is correct declaration of function pointer fp to a function which takes constant pointer to an integer as parameter and returns integer?

a. `int *fp (const int * intP)`
b. `int (*fp) (const int * intP)`
c. `int *fp (int const * intP)`
[✓ d.] `int (*fp) (int const* intP)`

Q5. What is the output of the following program:

```
void main()
```