به‌نام خدای مهربانی‌ها

دوره‌ی آموزشی المپیاد کامپیوتر

آزمون پایان‌دوره‌ای درس برنامه‌نویسی C/C++

پنج‌شنبه ۶ مردادماه ۱۳۹۰

نصیری‌شرق، شاه‌محمدی **مدّت آزمون: ۱۲۰ دقیقه**

بخش یکم) کوتاه پاسخ‌های شرور! ..... ۱۵ نمره

به پرسش‌های جواب مختصر، مفید و جامع بدهید.

1. **(۱ نمره)** می‌دانیم یک راه برای استفاده از cin این است که بعد از includeها، `using namespace std` داشته باشیم. دو روش جایگزین به‌جای استفاده از این عبارت طولانی بیان کنید.

2. **(۲ نمره)** در زبان C قدیمی، مفهوم private وجود نداشت. به‌نظر شما این کار چه ضعفی را برای برنامه‌نویسان داشت؟

3. **(۳ نمره)** می‌دانیم برحسب بودن یا نبودن const (دو حالت) و بودن یا نبودن & (این هم دو حالت) در نحوه‌ی دریافت پارامتر یک تابع، جمعاً ۴ = ۲ × ۲ حالت مختلف داریم. **الف)** برای هر یک از این ۴ حالت تفاوت آن با ۳ حالت دیگر را در یک خط بیان کنید. **ب)** اگر تنها فاکتور مهم برای ما سرعت باشد و پارامتری ارسالی در داخل تابع تغییر نیابد، کدام یک از این ۴ حالت بهتر است، چرا؟

4. **(۴ نمره)** فرض کنید خط زیر در یک کد دیده شده است.

```
int *&a = p;
```

الف) این متغیر به چه معناست؟ چه چیزی را نگه می‌دارد؟ چگونه می‌توان از آن استفاده کرد؟ کاربرد آن کجاست؟

ب) اگر متغیر p در خط قبلی این خط تعریف شده باشد و تنها این دو متغیر در برنامه باشند، نوع p آن چیست؟

5. **(۵ نمره)** کد ۷ خطی زیر را در نظر بگیرید. این کد را اگر همین‌الآن کامپایل کنیم عبارت `World` را چاپ می‌کند. آیا می‌توانید بدون تغییر در تابع `main()` کاری کنید که در خروجی عبارت `Hello World` چاپ شود؟ (تغییری که به برنامه می‌دهید باید تا حد امکان کمترین تعداد کاراکتر را داشته باشد).

```
1. #include <iostream>
1. using namespace std;
2. // اینجا کد بنویسید
3. int main() {
4.    cout << "World" << endl;
5.    return 0;
6. }
```

پرسش دوم) مرتب‌سازی ..... ۱۰ نمره

علی یک struct سنگین به نام `heavy` تعریف کرده که در آن یک متغیر `long long`، ۳ متغیر `double *`، ۶ متغیر `int`، سه متغیر `char` و ۱۱ تا متغیر `char *` موجود است.

1. **(۲ نمره)** فرض کنید h یک متغیر از نوع این struct است. چگونه می‌توانیم حجم اشغال شده توسط h را (به بایت) پیدا کنیم؟ این مقدار چند است؟

می‌دانیم vector همیشه یک حافظه‌ی داخلی پویا دارد که تعداد عناصر رزرو شده‌ی آن توانی از ۲ است. این اندازه با تابع داخلی capacity() (نظیر cout << v.capacity() << endl;) قابل مشاهده است. می‌دانیم در صورتی که v.size() برابر با v.capacity() باشد و عنصر جدیدی را بخواهیم push_back کنیم، ابتدا یک آرایه‌ی پویای جدید به‌اندازه‌ی دو براب ر آرایه‌ی داخلی کنونی، درخواست (new) می‌شود. سپس عناصر موجود در آرایه‌ی قبلی به نیمه‌ی اوّل این آرایه نقل‌مکان می‌کنند.

2. **(۲ نمره)** اگر با شروع از یک vector<heavy> خالی، ما ۲۰۰ شیء از ساختار heavy را دانه دانه در vectorمان، push_back کنیم، تعداد متوسط نقل‌مکان‌های هر عنصر vector چندتا است؟

3. **(۳ نمره)** اکنون فرض کنید می‌خواهیم عناصر این vector را بر حسب مقدارِ تنها متغیرِ long long موجود در هر کدام از عناصر، به‌صورت نزولی، مرتب کنیم. روش انجام این کار با استفاده‌ی الزامی از تابع sort را به همراه کدهای مربوطه بنویسید.

4. **(۳ نمره)** می‌دانیم تابع sort در هنگام مرتب‌سازی، بارها عناصر را بین اندیس‌های مختلف vector دوبه‌دو جابه‌جا (swap) می‌کند. این کار با توجه به حجم نسبتاً بالای هر عنصر از این struct، زمان‌بر است. با این وصف، چگونه می‌توان سرعت انجام این عمل مرتب‌سازی را بهبود ببخشیم؟ (بهینه‌سازی‌های بدیهی که ممکن‌ست کامپایلر برای اُپتیمایز کردن انجام دهد، مدّ نظر نیستند.)

پرسش سوم) درخت دودویی جستجو (BST) بسازیم ..... ۲۵ نمره

برای ساخت و نگه‌داری یک درخت دودودیی (Binary Search Tree) ساختار زیر به ما داده شده است و ما اجازه نداریم در آن تغییری به‌وجود بیاوریم.

```
1. struct node {
2.   node *left, *right;
3.   int value;
4.   node(){
5.     right = left = value = 0;
6.   }
7. };
```

برای این ساختار کدهای زیر را با حداقل تعداد حلقه و شرط، در زیبایی کامل بنویسید.

1. **(۲ نمره)** برای ساختار بالا عملگرهای > و < را طوری **بازتعریف** کنید که بر حسب مقدار valueی دو گره به‌درستی کار کنند.
2. **(۳ نمره)** تابع node *insert(node *root, int x) را بنویسید که گره‌ای با مقدار x را در زیر درخت به-ریشه‌ی root درج کند. این عمل باید ساختار درخت را (همه نوادگان راست، ناکوچکتر و همه نوادگان چپ، نابزرگتر از ریشه) حفظ کند.
3. **(۳ نمره)** تابع node *find(node *root, int x) را بنویسید که در زیردرخت به‌ریشه‌ی root، عنصر x را بیابد. اگر موجود بود اشاره‌گری به آن گره برگرداند (اگر چند تا بود، هر کدام‌شان مقبول است). اگر موجود نبود هم باید 0 یا NULL برگرداند.
4. **(۳ نمره)** تابع void delete(node *here) را بنویسید که گره‌ی پارامتر را حذف کند.
5. **(۵ نمره)** تابع void print(node *root, int mode) را بنویسید که عناصر زیردرخت را در یک خط چاپ کند. برحسب این که mode برابر با صفر یا یک یا دو باشد، خروجی می‌بایست preorder یا inorder یا postorder باشد. تعداد دستورهای شما (با شمارش معقول و نه چپاندن!) می‌بایست حداقل باشد.
6. **(۳ نمره)** تابع node *next(node *here) را بنویسید که گره‌ی حاوی کوچکترین عدد بزرگتر از here->value را برگرداند. در این آیتم فرض کنید اعداد همگی متفاوتند. اگر here->value عدد بیشینه بود باید اشاره گری به 0 (همان NULL) برگردانده شود.
7. **(۳ نمره)** تابع bool checktree(node *root) را بنویسید که بررسی کند زیردرخت به‌ریشه‌ی root از نظر نسبت مقادیر موجود در گره‌ها آیا معتبر است یا نه؟ به این معنی که آیا مقادیر گره‌های درخت ساختار BST را دارند یا نه؟

8. (**۳ نمره**) برای هر کدام از ۶ تابع بالا مشخص کنید که پارامتر اشاره‌گری که تابع مربوطه می‌گیرد، آیا می‌تواند const باشد یا نه؟

بخش چهارم) اندیس‌های محبوب ..... ۱۵ نمره

1. برنامه زیر را در نظر بگیرید:

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int main() {
5.      int n;
6.      cin >> n; // n is always less than 20
7.      int a[20];
8.      for(int i=0; i<n; ++i)
9.        cin >> a[i];
10.     // اینجا کد بنویسید
11.     // ...
12.     // ...
13.     Return 0;
14. }
```

در قسمت نشان داده شده، دستوراتی (نه لزوماً فقط ۳ خط!) بنویسید که تنها با استفاده از دو دستور حلقه و دو دستور شرط، بررسی کند که آیا اندیس‌های $b_1, b_2, \dots, b_k$ وجود دارند به طوری که $a[b_1] \wedge a[b_2] \wedge \dots \wedge a[b_{k-1}] = a[b_k]$ (همه b_i ها باید متمایز باشند و $2 \le k \le n \le 20$، ترتیب b_i ها مهم نیست). در صورت وجود این اعداد، خروجی برنامه شما باید برابر YES و در غیر این صورت برابر NO باشد.

بخش پنجم) خطای کامپایل، وای، وای! ..... ۱۰ نمره

خطاهای و اخطار (warning)های زمان کامپایل برنامه زیر را پیدا کنید و برای هر یک، توضیحی مختصر در مورد علت ایجاد خطا و شیوه بر طرف کردن آن بنویسید. کامپایلر مورد استفاده همان کامپایلر کلاس عملی است.

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  struct interval {
5.      public:
6.      int start = 0, end = 0;
7.
8.      private:
9.      int length()
10.     {
11.       return end - start;
12.     }
13.
14.     public:
15.     bool operator < (const interval &x) const
16.     {
17.       if(this->length() != x.length())
18.             return this->length() < x.length();
19.       return this->start < x.start;
20.     }
21. }
22.
23. int main() {
24.     vector<interval> v;
25.     v.resize(5);
26.     for(int i=0; i<5; ++i)
27.       cin >> v[i].start >>  v[i].end;
28.     sort(a, a+5);
29.     for(int i=0; i<5; ++i)
30.       cout << v[i]->start << " " << v[i]->end << endl;
```

```
31. }
```

بخش ششم) باگ‌گیری، واه، واه! ..... ۲۵ نمره

فرض کنید هر یک از برنامه‌های زیر در داخل یک تابع ()main با includeهای مناسب نوشته شده‌اند. خروجی هر برنامه را بنویسید. اعداد ابتدای خط شماره خط هستند و تنها برای ارجاع راحت‌تر شما قرار داده شده‌اند.

در صورتی که هر کدی خطای کامپایل/اجرا دارد آن را ذکر کرده، برطرف کرده، و سپس نتیجه رو بنویسید.

برنامه شماره یک (۳ نمره):

```
1.   int x = 5 , *y = new int (10);
2.   *y += 1;
3.   x += *y;
4.   y = &x;
5.   *y += 10;
6.   cout << x << " "<< y << endl;
```

برنامه شماره دو (۳ نمره):

```
1.   int x = 0XDEADBEEF;
2.   // A Dead Beef Tastes Bitter!
3.   int y = 010;
4.   int z = x % y;
5.   int w = 0XBAD;
6.   cout <<z<<" "<<(w&-1)<< endl;
```

برنامه شماره سه (۵ نمره):

```
1.   int c = 0, d = 0;
2.   for (int i=0; i<010; i++){
3.     for (int j=0; j<8; j++)
4.       if ((i|j) == (i^j))
5.         c++;
6.         d += c*c;
7.   }
8.   cout << d << endl;
```

برنامه شماره چهار (۵ نمره):

```
1.   char s[10];
2.   strcpy(s, "bcdefg");
3.   // a = (1100001) in ASCII
4.   for(int i=1; i<strlen(s); i++){
5.     s[i] |= s[i-1];
6.     cout << s[i];
7.     s[i+1] ^= s[i];
8.   }
9.   cout << endl;
```

برنامه شماره پنج (۴ نمره):

```
1.   struct st{
2.     char c;
3.     st(int v=0x0) {c=v^v;};
4.   } ar[12];
5.   for (int i=0; i<10; i++)
6.     ar[i].c = 'A'|i;
7.   char *p = new char[12];
8.   p = (char *)&0x0[ar];
9.   cout << p << " " << *p << endl;
```

برنامه شماره شش (۵ نمره):

```
1.  vector<char *> v;
2.  bool f(int d, char *p) {
3.    v.push_back(p);
4.    return d?f(--d,++p):0;
5.  }
6.
7.  int main() {
8.    char s[] = "welcome";
9.    f(5,s);
10.   sort(v.begin(),v.end());
11.   v[0][2] = 'h';
12.   cout << v[2] << endl;
13.   return 0;
14. }
15.
```

«شاد و سربلند و سرفراز و جهانی باشید!»