بسم الله الرحمن الرحیم

تکلیف سری اول

درس هوش مصنوعی

تاریخ تحویل: ۹ آبان

دکتر فلسفین

پاییز ۹۹

لطفاً پیش از حل سوالات به موارد زیر دقت شود:

- تکلیف شامل ۵ سوال تئوری و ۱ سوال عملی می‌باشد که در آن باید با توجه به شماره‌ی دانشجویی خود اقدام به پیاده‌سازی دو الگوریتم جست‌وجوی محلی برای دو مسئله‌ی بهینه‌سازی نمایید.
- پس از تصحیح و ارزیابی کدها ممکن از شما درخواست شود در یک جلسه‌ی اسکایپی در رابطه با کد توضیح دهید. لذا لازم است به تمام قسمت‌های کد خود مسلط باشید.
- پاسخ سوالات تئوری را به فرمت pdf آماده و به همراه فایل کدهای خود فشرده کرده و در سامانه در بخش مربوط به تکلیف اول آپلود نمایید.
- در تحویل تکلیف به زمان مجاز تعیین شده در سامانه برای آپلود پاسخ‌ها دقت فرمایید. پس از این زمان به هیچ طریقی تکلیف دریافت نشده و مورد بررسی قرار نمی‌گیرند.
- پاسخ تکالیف خود را حتما در سامانه آپلود کیند و از ارسال فایل پاسخ به ایمیل یا تلگرام اکیدا خودداری نمایید.
- در صورت وجود یا بروز هرگونه ابهام در سولات می‌توانید از طریق ایمیل زیر با TA درس در ارتباط باشید.

arashmarioriyad@gamil.com

سوال اول)

فرض کنید یک الگوریتم ژنتیک از کروموزوم‌هایی به فرم $x = abcdefgh$ با طول ثابت ۸ ژن استفاده می‌کند. هر ژن می تواند یک عدد بین ۰ تا ۹ باشد و برازندگی هر کروموزوم x به صورت زیر محاسبه شود:

$$f(x) = (a + b) - (c + d) + (e + f) - (g + h)$$

که هر چه مقدار تابع f بیشتر باشد، به معنای برازندگی بیشتر کروموزوم مربوطه است. همچین فرض کنید که جمعیت اولیه از ۴ فرد با کروموزوم‌های زیر تشکیل شده باشد:

$$x_1 = 65413532$$
$$x_2 = 87126601$$
$$x_3 = 23921285$$
$$x_4 = 41852094$$

با توجه به موارد مطرح شده به سوالات زیر پاسخ دهید:

الف- برازندگی هر فرد را تعیین و آنها را به ترتیب از بیشترین به کمترین برازندگی مرتب کنید.

ب- عملیات‌های crossover زیر را انجام دهید (برای آگاهی بیشتر از انواع مختلف عملیات crossover مطرح شده در این سوال می‌توانید به این لینک مراجعه نمایید).

- دو تا برازنده‌ترین افراد را با تقاطع یک نقطه (single point crossover) ترکیب کنید .
- دومین و سومین فرد را با تقاطع دو نقطه (two point crossover) ترکیب کنید.
- اولین و سومین فرد را با تقاطع یکنواخت (uniform crossover) ترکیب کنید

پ- جمعیت حاصل از قسمت ب شامل ۶ فرزند می‌باشد که از عملیات تقاطع گرفته شده‌اند. با فرض آنکه می‌خواهیم اندازه‌ی جمعیت ثابت بماند، کروموزوم‌های با برازندگی کمتر را مشخص کرده و کنار بگذارید تا جمعیت جدید با اندازه‌ی ۴ از برازنده‌ترین افراد به دست آید. برازندگی کل جمعیت جدید را نسبت به جمعیت اولیه مقایسه کنید. آیا مقدار برازندگی کل بهبود یافته‌است؟ (برازندگی کل یک جمعیت را برابر با مجموع برازندگی اعضای آن جمعیت می‌دانیم.)

ت- با توجه به تابع برازندگی و با در نظر گرفتن این که ژن‌ها می توانند فقط اعدادی بین ۰ تا ۹ باشند، کروموزومی را پیدا کنید که نشان دهنده ی جواب بهینه (جواب دارای بالاترین برازندگی) باشد. مقدار بیشترین برازندگی را پیدا کنید.

ث- برای نمایش مطرح شده در سوال، یک عملگر جهش مناسب معرفی نمایید.

ج- با نگاه کردن به جمعیت اولیه‌ی الگوریتم و با استفاده از عمگر تقاطع یک نقطه (single point crossover) آیا می توانید تعیین کنید که این جمعیت بدون عملگر جهش می‌تواند به جواب بهینه سراسری برسد یا خیر؟

سوال دوم)

با ذکر دلیل مشخص کنید هریک از الگوریتم های زیر با شرایط مشخص شده به چه الگوریتم جست‌وجوی محلی‌ای تبدیل می‌شوند:

الف- الگوریتم شبیه سازی ذوب فلزات با $T = 0$

ب- الگوریتم شبیه سازی ذوب فلزات با $T = +\infty$

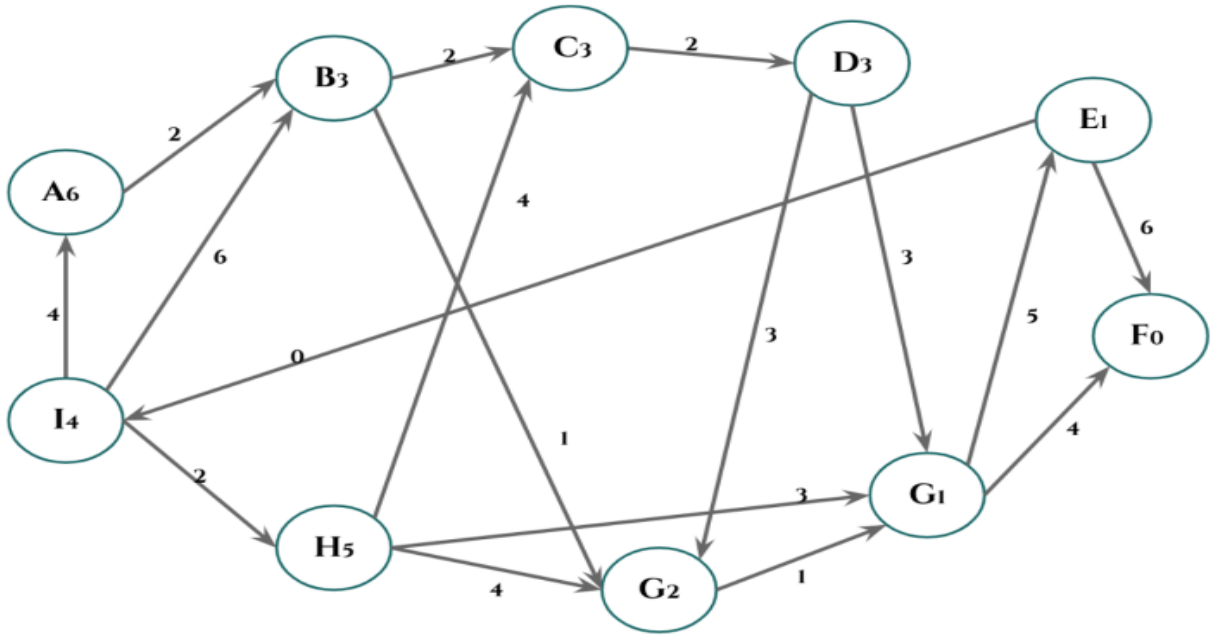
پ- الگوریتم جست‌وجوی پرتوی محلی با $K = 1$

ت- الگوریتم جست‌وجوی پرتوی محلی با $K = +\infty$

ث- الگوریتم ژنتیک وقتی که هر نسل فقط شامل یک نفر باشد

سوال سوم)

گراف ساده و جهت‌دار زیر را در نظر بگیرید. A گره شروع، F گره هدف و اعداد داخل هر گره هزینه تخمینی هر گره با هدف و اعداد روی یال میزان پاداش برای انتقال از گره فعلی به گره بعدی است.



با توجه به گراف معرفی شده، به سوالات زیر پاسخ دهید.

الف- الگوریتم Hill Climbing یک بار با هدف یافتن کمترین هزینه و بار دیگر با هدف یافتن بیشترین پاداش چه مسیرهایی را به عنوان پاسخ می‌یابد؟

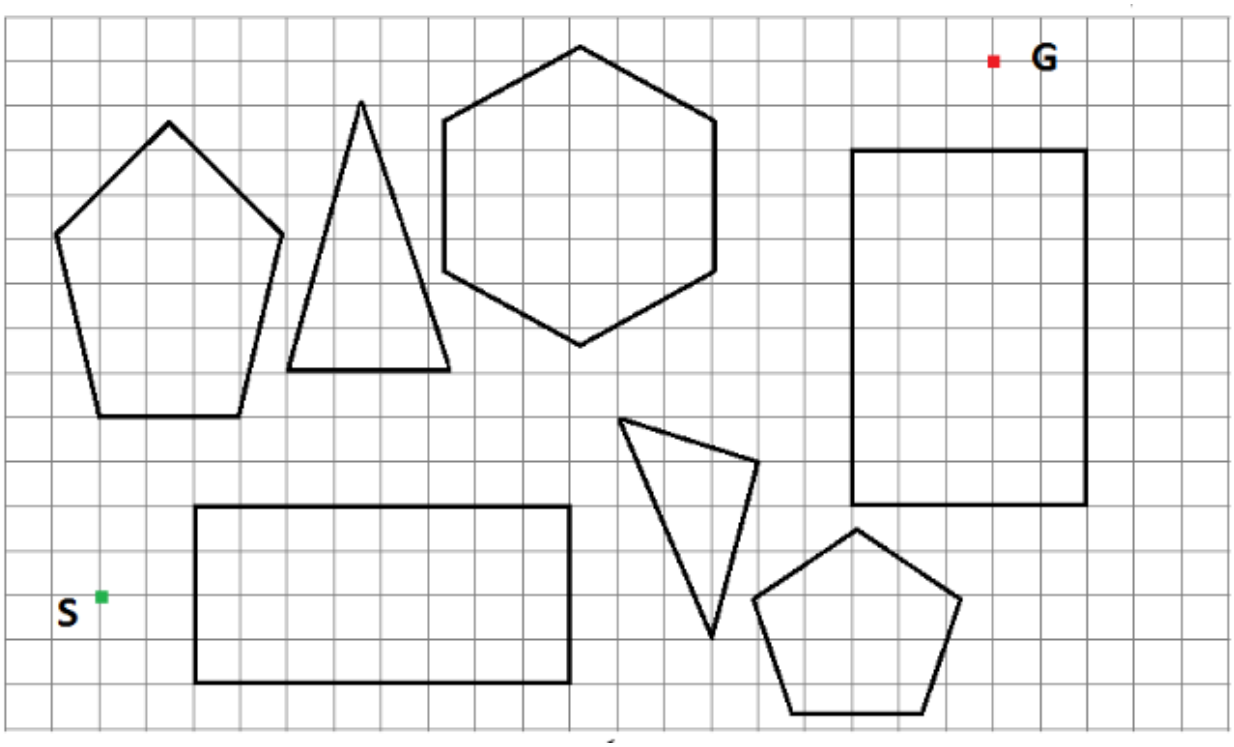
ب- الگوریتم Tabu Search یک بار با هدف یافتن کمترین هزینه و بار دیگر با هدف یافتن بیشترین پاداش چه مسیرهایی را به عنوان پاسخ می‌یابد؟

پ- مشخص کنید که هر کدام از الگوریتم‌ها در ماکزیمم محلی متوقف شده‌اند یا در ماکزیمم سراسری؟

ت- آیا با استفاده از این دو الگوریتم می‌توان مسیری را یافت که همزمان بیشترین پاداش و کمترین هزینه را داشته‌باشد؟ در صورت منفی بودن جواب آیا می‌توانید الگوریتمی مبتنی بر جست‌وجوی محلی پیشنهاد دهید که مسیری با بیشترین پاداش و کمترین هزینه را بیابد؟

سوال چهارم)

مسئله یافتن کوتاه‌ترین مسیر بین دو نقطه در یک محیط با موانع چند ضلعی محدب مانند شکل زیر را در نظر بگیرید.



این مسئله شبیه مسئله‌ای است که در آن یک ربات برای حرکت در یک محیط شلوغ و همراه با موانع با آن مواجه است. فرض کنید که نقطه ی شروع S، نقطه‌ی هدف G و فضای حالت مسئله تمام نقاط با مختصات صحیح در صفحه باشد.

با توجه به شکل فوق به سوالات زیر پاسخ دهید:

الف- چگونگی کار الگوریتم تپه‌نوردی برای رسیدن به نقطه‌ی هدف را شرح دهید.

ب- با یک مثال نشان دهید چگونه ممکن است ربات در یک محیط با موانع غیر محدب در بهینه محلی گیر کند.

پ- آیا در محیطی با موانع محدب امکان گیر کردن در بهینه محلی وجود دارد؟

ت- آیا استفاده از الگوریتم شبیه‌سازی ذوب فلزات در اینگونه از مسائل همیشه باعث فرار از بهینه محلی می‌شود؟ جواب خود را توضیح دهید.

سوال پنجم)

مسئله‌ی خوشه‌بندی گراف را در نظر بگیرید.

ورودی: یک گراف ساده، بدون جهت و وزن‌دار - عدد طبیعی k

خروجی: افراز رئوس گراف به k خوشه (دسته) به گونه‌ای که مقدار روبه‌رو بیشینه شود: $f = W_1 - W_2$

که در آن W_1 مجموع وزن یال‌هایی می‌باشد که دو سر آن‌ها (دو راس انتهایی آن یال) درون یک خوشه قرار گرفته‌اند و W_2 مجموع وزن یال‌هایی می‌باشد که دو سر آن‌ها در خوشه‌های متفاوتی قرار دارند.

می‌خواهیم برای این مسئله با استفاده از الگورتیم تپه‌نوردی یک بستر جست‌وجو در فضای مسئله طراحی کنیم که به دنبال بیشینه کردن مقدار تابع f باشد.

الف- یک representation مناسب برای معرفی هر یک از اعضای فضای جست‌وجوی مسئله معرفی نمایید.

ب- با توجه به representation معرفی شده توسط شما در قسمت قبل، تعداد کل اعضای فضای جست‌وجو را مشخص کنید. (فرض کنید گراف ورودی n راس و m یال دارد)

پ- یک تعریف همسایگی مناسب برای فضای جست‌وجو ارائه دهید.

ت - با توجه به تعریف همسایگی ارائه شده، بیشینه و کمینه تعداد همسایه‌های اعضای فضای جست‌وجو را مشخص کنید.

ث- بررسی نمایید آیا برای این مسئله الگوریتم دقیق چند جمله‌ای وجود دارد یا خیر. در صورت وجود آن را معرفی کرده و پیچیدگی الگوریتم مطرح شده را بررسی نمایید.

سوال ششم)

در این سوال قصد داریم برای حل مسائل معروفی از دنیای علوم کامپیوتر، از الگوریتم‌های غیردقیق و مبتنی بر جست‌وجوی محلی استفاده نماییم. بدین‌ترتیب که هر دانشجو باید با توجه به جدول شماره‌ی ۱ و شماره‌ی دانشجویی خود، دو مسئله‌ی بهینه‌سازی را با استفاده از الگوریتم‌های مشخص شده در جدول حل و با استفاده از یک زبان برنامه‌نویسی پیاده‌سازی نماید.

الگوریتم‌ها/مسائل	Random Restart Hill Climbing	Tabu Search	Simulated Annealing	Genetic Algorithm
Max-SAT	• 9526193 • 9629163 • 9631983 • 9727013	• 9530463 • 9630513 • 9633003 • 9728043	• 9531663 • 9629373 • 9631243 • 9734943 • 9737483	• 9627163 • 9632463 • 9635973 • 9636533
Symmetric TSP	• 9626633 • 9629743 • 9637353 • 9733313	• 9528403 • 9627993 • 9701173 • 9735653	• 9529053 • 9626903 • 9727783 • 9731843	• 9624193 • 9624923 • 9725113 • 9735123
0-1 Knapsack	• 9624923 • 9629373 • 9632463 • 9728043	• 9531663 • 9626633 • 9629163 • 9735123	• 9526193 • 9627993 • 9633003 • 9637353	• 9629743 • 9631983 • 9731843 • 9734943
Graph Coloring	• 9528403 • 9624193 • 9631243 • 9636533	• 9529053 • 9635973 • 9727783 • 9733313	• 9627163 • 9630513 • 9701173 • 9725113	• 9530463 • 9626903 • 9727013 • 9735653 • 9737483

جدول-۱

الگوریتم‌های مطرح شده در جدول ۱ همگی در کلاس درس معرفی و مورد بررسی قرار گرفته‌اند و تعریف مسائل جدول فوق را می‌توانید در جدول شماره‌ی ۲ مشاهده نمایید.

نام مسئله	تعریف مسئله
MAX-SAT	• ورودی: یک عبارت منطقی به فرم CNF • خروجی: یافتن بیشینه تعداد کلاوزهایی که می‌توانند با مقداردهی مناسب به لیترال‌ها، دارای ارزش True شوند.
Symmetric TSP	• ورودی: یک گراف کامل، همبند، ساده، بدون جهت و وزن‌دار • خروجی: یافتن دور همیلتنی با کمترین وزن (دوری که از یک راس شروع شده و از تمام رئوس دقیقا یک بار عبور کند و به راس نخست باز گردد و در عین حال مجموع وزن یال‌های متناظر با آن دور کمینه باشد.) بیان دیگری از این مسئله بدین شکل است که یک فروشنده قصد دارد در محیطی شامل تعدادی شهر، سفر خود را از یک شهر آغاز کرده و با تنها یک بار عبور کردن از هر شهر در نهایت به شره آغازین باز گردد. در عین حال باید تلاش شود که طول مسیر طی شده در این سفر کمینه شود.
0-1 Knapsack	• ورودی: تعداد n آیتم با وزن w_i و ارزش v_i برای $1 \leq i \leq n$ و یک کوله‌پشتی با ظرفیت وزنی W • خروجی: مجموعه‌ای از آیتم‌ها که مجموع وزن آن‌ها از ظرفیت کوله‌پشتی تجاوز نکند و مجموع ارزش آن‌ها بیشینه باشد.
Graph Coloring	• ورودی: یک گراف ساده و بدون جهت • خروجی: یافتن کمترین تعداد رنگی که که بتوان با استفاده از آن رنگ‌ها رئوس گراف را به گونه‌ای رنگ‌آمیزی کرد که هیچ دو راس مجاوری در گراف رنگ یکسان نداشته‌باشند.

جدول-۲

فرمت ورودی و خروجی نمونه‌های هر مسئله نیز در جدول شماره‌ی ۳ مشخص شده‌است.

نام مسئله	فرمت ورودی	فرمت خروجی
MAX-SAT	خط اول ورودی شامل دو عدد و به‌ترتیب مشخص کننده‌ی تعداد متغیرها و تعداد کلاوزها است. خط‌های بعدی (به تعداد کلاوزها) هر کدام مشخص‌کننده‌ی شماره‌ی متغیرهای حاضر در کلاوز مربوطه است (علامت منفی نشان‌دهنده‌ی عملگر not می‌باشد و هر خط با عدد ۰ به پایان می‌رسد). دقت شود که متغیرها از شماره‌ی ۱ شروع به نام‌گذاری شده‌اند.	در خط اول یک عدد طبیعی چاپ کنید که نشان‌دهنده‌ی بیشینه تعداد کلاوزهایی است که مطابق الگوریتم شما ارزش True دارند. در خطوط بعدی (به تعداد متغیرها) در هر خط یکی از دو عدد ۰ یا ۱ را چاپ نمایید که به ترتیب نشان‌دهنده‌ی آن است که آیا متغیر مربوطه (بر اساس شماره‌ی متغیر به صورت صعودی) مقدار False گرفته است یا مقدار True

Symmetric TSP	خط اول شامل یک عدد طبیعی است که نشان‌دهنده‌ی تعداد شهرها (رئوس گراف) می‌باشد. خط‌های بعدی (به تعداد شهرها) شامل ۳ عدد است که به ترتیب شماره‌ی شهر و مختصات شهر (دو بعدی) می‌باشند.	در خط اول یک عدد چاپ کنید که نشان‌دهنده‌ی مجموع وزن یال‌های مشارکت‌کننده در دور همیلتونی حاصل از الگوریتم بیشینه‌سازی شماست. (مجموع فاصله‌ی میان جفت شهرهای انتخاب شده در دور همیلتنی حاصل از الگوریتم شما) در خط دوم یک دنباله از اعداد (به تعداد شهرها) چاپ کنید که در واقع نشان‌دهنده‌ی ترتیب شماره‌ی شهرها در دور همیلتنی به‌دست آمده از الگوریتم شماست.
0-1 Knapsack	خط اول شامل دو عدد است که به ترتیب نشان‌دهنده‌ی تعداد آیتم‌ها و وزن قابل تحمل برای کوله‌پشتی می‌باشند. خط‌های بعدی (به تعداد آیتم‌ها) هر کدام شامل دو عدد است که به‌ترتیب نشان‌دهنده‌ی ارزش و وزن آیتم‌هاست.	در خط اول خروجی دو عدد چاپ نمایید که به ترتیب نشان‌دهنده‌ی مجموع ارزش و مجموع وزن آیتم‌های انتخاب شده توسط الگوریتم شماست. در خطوط بعدی (به تعداد آیتم‌ها) یک عدد از میان ۰ و ۱ چاپ نمایید که به ترتیب نشان‌دهنده‌ی حضور یا عدم حضور آیتم (بر اساس ترتیب آیتم‌ها در ورودی) در مجموعه‌ی جواب نهایی حاصل از الگوریتم شماست.
Graph Coloring	خط اول شامل دو عدد می‌باشد که به ترتیب نشان‌دهنده‌ی تعداد رئوس و یال‌ها در گراف است. خط‌های بعدی (به تعداد یال‌ها) ابتدا شامل کاراکتر e است که معرف یال می‌باشد (به این کاراکتر توجهی نشود) و سپس دو عدد ظاهر می‌شود که در واقع شماره‌ی رئوس دو سر یال مربوطه هستند (شماره‌ی رئوس از ۱ آغاز می‌شود)	در خط اول خروجی یک عدد طبیعی چاپ کنید که همان کمینه تعداد رنگ‌های لازمی است که الگوریتم شما برای برای رنگ‌آمیزی گراف پیشنهاد کرده است. در خطوط بعدی (به تعداد رئوس) در هر خط دو عدد چاپ نمایید که عدد اول مشخص‌کننده‌ی شماره‌ی راس و عدد دوم نشان‌دهنده‌ی شماره‌ی رنگ می‌باشد. (اگر الگوریتم شما عدد k را به عنوان تعداد زنگ‌ها در نظر بگیرد، شماره‌ی رنگ‌ها از ۱ تا k خواهد بود.)

جدول-۳

همچنین برای هر مسئله در این تکلیف دو نمونه فایل ورودی برای شما فراهم شده است که اطلاعات آن را می‌توانید در جدول شماره‌ی ۴ مشاهده نمایید. در ادامه نیز سعی می‌شود جواب دقیق نمونه‌های موجود در جدول ۴ در اختیار شما قرار گیرد تا با توجه به آن‌ها دقت و کارایی الگوریتم خود را محک بزنید.

نام مسئله	نام نمونه‌ی ۱	نام نمونه‌ی ۲
MAX-SAT	max-sat-20-80.txt	max-sat-20-90.txt
Symmetric TSP	tsp-52.txt	tsp-100.txt
0-1 Knapsack	knapsack-20.txt	knapsack-100.txt
Graph Coloring	Graph-coloring-11-20.txt	Graph-coloring-23-71.txt

جدول-۴

در رابطه با این سوال لطفا به موارد زیر توجه نمایید:

- برای پیاده‌سازی الگوریتم‌ها می‌توانید از تمامی زبان‌های برنامه‌نویسی مرسوم (مانند c/c++، پایتون، جاوا و ...) استفاده نمایید.
- اسم دو فایل کد خود را به صورت اسم مسئله‌ی مورد نظر در کنار اسم الگوریتم خود نام‌گذاری کنید. (به صورت problem-algorithm)
- اگر اجرای صحیح کد شما نیازمند توجه به نکاتی می‌باشد، حتما در قالب یک فایل pdf موارد لازم را توضیح دهید.
- کد خود را به گونه‌ای بنویسید که با دریافت نام یا مسیر فایل ورودی، آن فایل را بخواند و الگوریتم را اجرا نماید.
- کد شما روی تعدادی نمونه بررسی شده و با توجه به میزان نزدیکی جواب الگوریتم شما به جواب دقیق مسئله، نمره‌دهی و ارزیابی صورت می‌پذیرد.
- به هر کد شما حداکثر ۱ دقیقه فرصت داده می‌شود تا جواب نهایی خود را ارائه دهد. لذا در الگوریتم Random Restart Hill Climbing تعداد نقاط شروع تصادفی و در الگوریتم ژنتیک اندازه‌ی جمعیت را به گونه‌ای تنظیم نمایید که مدت زمان اجرای الگوریتم شما از ۱ دقیقه تجاوز ننماید.
- در صورتی که کد شما دچار خطا شود یا به هر دلیلی به صورت کامل اجرا نشود، نمره‌ای به این بخش تعلق نمی‌گیرد (طبعاً اگر جواب نهایی الگوریتم شما نامعتبر باشد، مثلا بهتر از جواب دقیق مسئله باشد، نمره‌ای تعلق نمی‌گیرد).

موفق باشید